

# On the Timed Temporal Logic Planning of Coupled Multi-Agent Systems

Alexandros Nikou<sup>a</sup>, Dimitris Boskos<sup>a</sup>, Jana Tumova<sup>b</sup> and Dimos V. Dimarogonas<sup>a</sup>

<sup>a</sup>The authors are with KTH Center of Autonomous Systems and ACCESS Linnaeus Center, School of Electrical Engineering, KTH Royal Institute of Technology, SE-100 44, Stockholm, Sweden.

<sup>b</sup>The author is with School of Computer Science and Communication, KTH Royal Institute of Technology, SE-100 44, Stockholm, Sweden.

## ARTICLE HISTORY

Compiled October 17, 2017

## ABSTRACT

This paper presents a fully automated procedure for controller synthesis for multi-agent systems under coupling constraints. Each agent is modeled with dynamics consisting of two terms: the first one models the coupling constraints and the other one is an additional bounded control input. We aim to design these inputs so that each agent meets an individual high-level specification given as a Metric Interval Temporal Logic (MITL). First, a decentralized abstraction that provides a space and time discretization of the multi-agent system is designed. Second, by utilizing this abstraction and techniques from formal verification, we propose an algorithm that computes the individual runs which provably satisfy the high-level tasks. The overall approach is demonstrated in a simulation example conducted in MATLAB environment.

## KEYWORDS

Multi-Agent Systems, Cooperative Control, Hybrid Systems, Formal Verification, Timed Logics, Abstractions, Discrete Event Systems.

## 1. Introduction

Cooperative control of multi-agent systems has traditionally focused on designing distributed control laws in order to achieve global tasks such as consensus and formation control, and at the same time fulfill properties such as network connectivity and collision avoidance. Over the last few years, the field of control of multi-agent systems under high-level specifications has been gaining attention. In this work, we aim to additionally introduce specific time bounds into these tasks, in order to include specifications such as: “Robot 1 and robot 2 should visit region  $A$  and  $B$  within 4 time units respectively or “Both robots 1 and 2 should periodically survey regions  $A_1$ ,  $A_2$ ,  $A_3$ , avoid region  $X$  and always keep the longest time between two consecutive visits

---

This work was supported by the H2020 ERC Starting Grant BUCOPHSYS, the EU H2020 Research and Innovation Programme under GA No. 731869 (Co4Robots), the SSF COIN project, the Swedish Research Council (VR) and the Knut och Alice Wallenberg Foundation.

CONTACT Alexandros Nikou. Email: anikou@kth.se, Dimitris Boskos. Email: boskos@kth.se, Jana Tumova. Email: tumova@kth.se and Dimos V. Dimarogonas. Email: dimos@kth.se

to  $A_1$  below 8 time units”.

The qualitative specification language that has primarily been used to express the high-level tasks is Linear Temporal Logic (LTL) (see, e.g., [1]). There is a rich body of literature containing algorithms for verification and synthesis of multi-agent systems under high level specifications ([2–4]). Controller synthesis under timed specifications has been considered in [5–8]. In [5], the authors addressed the problem of designing high-level planners to achieve tasks for switching dynamical systems under Metric Temporal Logic (MTL) specification and in [6], the authors utilized a counterexample-guided synthesis for cyber-physical systems subject to Signal Temporal Logic (STL) specifications. In [7], an optimal control problem for continuous-time stochastic systems subject to objectives specified in MITL was studied. In [8], the authors focused on motion planning based on the construction of an efficient timed automaton from a given MITL specification. However, all these works are restricted to single agent planning and are not extendable to multi-agent systems in a straightforward way. The high-level coordination of multiple vehicles under timed specifications has been considered in [9], by solving an optimization problem over the tasks’ execution time instances.

An automata-based solution for multi-agent systems was proposed in our previous work [10], where Metric Interval Temporal Logic (MITL) formulas were introduced in order to synthesize controllers such that every agent fulfills an individual specification and the team of agents fulfills a global specification. Specifically, the abstraction of each agent’s dynamics was considered to be given and an upper bound of the time that each agent needs to perform a transition from one region to another was assumed. Furthermore, potential coupled constraints between the agents were not taken into consideration. Motivated by this, in this work, we aim to address the aforementioned issues. We assume that the dynamics of each agent consists of two parts: the first part is a consensus type term representing the coupling between the agent and its neighbors, and the second one is an additional control input which will be exploited for high-level planning. Hereafter, we call it a free input. A decentralized abstraction procedure is provided, which leads to an individual Transition System (TS) for each agent and provides a basis for high-level planning. Additionally, this abstraction is associated to a time quantization which allows us to assign precise time durations to the transitions of each agent. Abstractions for both single and multi-agent systems can be found in [11–16]. Compositional frameworks are provided in [14] for safety specifications of discrete time systems, and [15], which is focused on feedback linearizable systems with a cascade interconnection. Therefore, these results are not applicable to the systems we consider, which evolve in continuous time and do not require a specific network interconnection.

Motivated by our previous work [16], we start from the consensus dynamics of each agent and we construct a Weighted Transition System (WTS) for each agent in a decentralized manner. Each agent is assigned an individual task given in MITL formulas. We aim to design the free inputs so that each agent performs the desired individual task within specific time bounds. In particular, we provide an automatic controller synthesis method for coupled multi-agent systems under high-level tasks with timed constraints. A motivation for this framework comes from applications such as the deployment of aerial robotic teams. In particular, the consensus coupling allows the robots to stay sufficiently close to each other and maintain a connected network during the evolution of the system. Additionally, individual MITL formulas are leveraged to assign area monitoring tasks to each robot individually. The MITL formalism enables us to impose time constraints on the monitoring process. Compared to exist-

ing works on multi-agent planning under temporal logic specifications, the proposed approach considers dynamically coupled multi-agent systems under timed temporal specifications in a distributed way. To the best of the authors' knowledge, this is the first time that a fully automated framework for multi-agent systems consisting of both constructing an abstraction and conducting high-level timed temporal logic planning is considered.

This paper is organized as follows. In Section 2 a description of the necessary mathematical tools, the notations and the definitions are given. Section 3 provides the dynamics of the system and the formal problem statement. Section 4 discusses the technical details of the solution. Section 5 is devoted to a simulation example. Finally, conclusions and future work are discussed in Section 6.

## 2. Notation and Preliminaries

### 2.1. Notation

Denote by  $\mathbb{R}, \mathbb{Q}_+, \mathbb{N}$  the set of real, nonnegative rational and natural numbers including 0, respectively. Define also  $\mathbb{T}_\infty = \mathbb{T} \cup \{\infty\}$  for a set  $\mathbb{T} \subseteq \mathbb{R}$ ;  $\mathbb{R}_{\geq 0}$  is the set of real numbers with all elements nonnegative. Given a set  $S$ , we denote by  $|S|$  its cardinality, by  $S^N = S \times \dots \times S$ , its  $N$ -fold Cartesian product and by  $2^S$  the set of all its subsets. For a subset  $S$  of  $\mathbb{R}^n$ , denote by  $\text{cl}(S)$ ,  $\text{int}(S)$  and  $\partial S = \text{cl}(S) \setminus \text{int}(S)$  its closure, interior and boundary, respectively, where  $\setminus$  is used for set subtraction. The notation  $\|x\|$  is used for the Euclidean norm of a vector  $x \in \mathbb{R}^n$  and  $\|A\| = \max\{\|Ax\| : \|x\| = 1\}$  for the induced norm of a matrix  $A \in \mathbb{R}^{m \times n}$ . Given a matrix  $A$  denote by  $\lambda_{\max}(A) = \max\{|\lambda| : \lambda \in \sigma(A)\}$  the spectral radius of  $A$ , where  $\sigma(A)$  is the set of all the eigenvalues of  $A$ ;  $A \otimes B$  denotes the Kronecker product of the matrices  $A, B \in \mathbb{R}^{m \times n}$  (see [17]). Define also by  $I_n \in \mathbb{R}^{n \times n}$  the identity matrix.

### 2.2. Multi-Agent Systems

An *undirected graph*  $\mathcal{G}$  is a pair  $(\mathcal{I}, \mathcal{E})$ , where  $\mathcal{I} = \{1, \dots, N\}$  is a finite set of nodes, representing a team of agents, and  $\mathcal{E} \subseteq \{\{i, j\} : i, j \in \mathcal{I}, i \neq j\}$ , is the set of edges that model the communication capability between the neighboring agents. For each agent, its neighbors' set  $\mathcal{N}(i)$  is defined as  $\mathcal{N}(i) = \{j_1, \dots, j_{N_i}\} = \{j \in \mathcal{I} : \{i, j\} \in \mathcal{E}\}$  where  $N_i = |\mathcal{N}(i)|$ . The Laplacian matrix  $L(\mathcal{G}) \in \mathbb{R}^{N \times N}$  of the graph  $\mathcal{G}$  is defined as  $L(\mathcal{G}) = D(\mathcal{G})D(\mathcal{G})^\top$  where  $D(\mathcal{G})$  is the  $N \times |\mathcal{E}|$  incidence matrix, as it is defined in [18, Chapter 2]. The graph Laplacian  $L(\mathcal{G})$  is positive semidefinite and symmetric. If we consider an ordering  $0 = \lambda_1(\mathcal{G}) \leq \lambda_2(\mathcal{G}) \leq \dots \leq \lambda_N(\mathcal{G}) = \lambda_{\max}(\mathcal{G})$  of the eigenvalues of  $L(\mathcal{G})$  then we have that  $\lambda_2(\mathcal{G}) > 0$  iff  $\mathcal{G}$  is connected ([18, Chapter 2]).

Given a vector  $x_i = (x_i^1, \dots, x_i^n) \in \mathbb{R}^n$ , the component operator  $c(x_i, \ell) = x_i^\ell \in \mathbb{R}, \ell = 1, \dots, n$  gives the projection of  $x_i$  onto its  $\ell$ -th component (see [18, Chapter 7]). Similarly, for the stack vector  $x = (x_1, \dots, x_N) \in \mathbb{R}^{Nn}$  the component operator is defined as  $c(x, \ell) = (c(x_1, \ell), \dots, c(x_N, \ell)) \in \mathbb{R}^N, \ell = 1, \dots, n$ . By using the component

operator, the norm of a vector  $x \in \mathbb{R}^{Nn}$  can be evaluated as  $\|x\| = \left\{ \sum_{\ell=1}^n \|c(x, \ell)\|^2 \right\}^{\frac{1}{2}}$ .

Denote by  $\tilde{x} \in \mathbb{R}^{|\mathcal{E}|n}$  the stack column vector of the vectors  $x_i - x_j, \{i, j\} \in \mathcal{E}$  with the edges ordered as in the case of the incidence matrix  $D(\mathcal{G})$ . Then, the following

holds:

$$\tilde{x} = \left( D(\mathcal{G})^\top \otimes I_n \right) x. \quad (1)$$

### 2.3. Cell Decompositions

In the subsequent analysis a discrete partition of the workspace into cells will be considered which is formalized through the following definition.

**Definition 1.** A *cell decomposition*  $S = \{S_\ell\}_{\ell \in \mathbb{I}}$  of a set  $\mathcal{D} \subseteq \mathbb{R}^n$ , where  $\mathbb{I} \subseteq \mathbb{N}$  is a finite or countable index set, is a family of uniformly bounded convex sets  $S_\ell, \ell \in \mathbb{I}$  such that  $\text{int}(S_\ell) \cap \text{int}(S_{\hat{\ell}}) = \emptyset$  for all  $\ell, \hat{\ell} \in \mathbb{I}$  with  $\ell \neq \hat{\ell}$  and  $\cup_{\ell \in \mathbb{I}} S_\ell = \mathcal{D}$ . We assume that the interiors of the cells are non-empty.

### 2.4. Time Sequence, Timed Run and Weighted Transition System

In this section we include some definitions from computer science that are required to analyze our framework.

An infinite sequence of elements of a set  $X$  is called an *infinite word* over this set and it is denoted by  $\chi = \chi(0)\chi(1)\dots$ . The  $i$ -th element of a sequence is denoted by  $\chi(i)$ . For certain technical reasons that will be clarified in the sequel, we will assume hereafter that  $\mathbb{T} = \mathbb{Q}_+$ .

**Definition 2.** ([19]) A *time sequence*  $\tau = \tau(0)\tau(1)\dots$  is an infinite sequence of time values  $\tau(j) \in \mathbb{T}$ , satisfying the following properties:

- Monotonicity:  $\tau(j) < \tau(j+1)$  for all  $j \geq 0$ .
- Progress: For every  $t \in \mathbb{T}$ , there exists  $j \geq 1$ , such that  $\tau(j) > t$ .

An *atomic proposition*  $p$  is a statement that is either True ( $\top$ ) or False ( $\perp$ ).

**Definition 3.** ([19]) Let  $AP$  be a finite set of atomic propositions. A *timed word*  $w$  over the set  $AP$  is an infinite sequence  $w^t = (w(0), \tau(0))(w(1), \tau(1))\dots$  where  $w(0)w(1)\dots$  is an infinite word over the set  $2^{AP}$  and  $\tau(0)\tau(1)\dots$  is a time sequence with  $\tau(j) \in \mathbb{T}, j \geq 0$ .

**Definition 4.** A *Weighted Transition System (WTS)* is a tuple  $(S, S_0, Act, \longrightarrow, d, AP, L)$  where  $S$  is a finite set of states;  $S_0 \subseteq S$  is a set of initial states;  $Act$  is a set of actions;  $\longrightarrow \subseteq S \times Act \times S$  is a transition relation;  $d : \longrightarrow \rightarrow \mathbb{T}$  is a map that assigns a positive weight to each transition;  $AP$  is a finite set of atomic propositions; and  $L : S \rightarrow 2^{AP}$  is a labeling function. The notation  $s \xrightarrow{\alpha} s'$  is used to denote that  $(s, \alpha, s') \in \longrightarrow$  for  $s, s' \in S$  and  $\alpha \in Act$ . For every  $s \in S$  and  $\alpha \in Act$  define  $\text{Post}(s, \alpha) = \{s' \in S : (s, \alpha, s') \in \longrightarrow\}$ .

**Definition 5.** A *timed run* of a WTS is an infinite sequence  $r^t = (r(0), \tau(0))(r(1), \tau(1))\dots$ , such that  $r(0) \in S_0$ , and for all  $j \geq 1$ , it holds that  $r(j) \in S$  and  $(r(j), \alpha(j), r(j+1)) \in \longrightarrow$  for a sequence of actions  $\alpha(1)\alpha(2)\dots$  with  $\alpha(j) \in Act, \forall j \geq 1$ . The *time stamps*  $\tau(j), j \geq 0$  are inductively defined as:

- (1)  $\tau(0) = 0$ .
- (2)  $\tau(j+1) = \tau(j) + d(r(j), \alpha(j), r(j+1)), \forall j \geq 1$ .

Every timed run  $r^t$  generates a *timed word*  $w(r^t) = (w(0), \tau(0)) (w(1), \tau(1)) \dots$  over the set  $2^{AP}$  where  $w(j) = L(r(j))$ ,  $\forall j \geq 0$  is the subset of atomic propositions that are true at state  $r(j)$ .

## 2.5. Metric Interval Temporal Logic

The syntax of *Metric Interval Temporal Logic (MITL)* over a set of atomic propositions  $AP$  is defined by the grammar:

$$\varphi := p \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \bigcirc_I \varphi \mid \diamond_I \varphi \mid \square_I \varphi \mid \varphi_1 \mathcal{U}_I \varphi_2,$$

where  $p \in AP$ , and  $\bigcirc, \diamond, \square$  and  $\mathcal{U}$  are the next, eventually, always and until temporal operator, respectively;  $I = [a, b] \subseteq \mathbb{T}$  where  $a, b \in [0, \infty]$  with  $a < b$  is a non-empty timed interval. MITL can be interpreted either in continuous or point-wise semantics [20]. In this paper, the latter approach is utilized, since the consideration of point-wise (event-based) semantics is more suitable for the automata-based specifications considered in a discretized state-space. The MITL formulas are interpreted over timed words like the ones produced by a WTS which is given in Def. 5.

**Definition 6.** ([20], [21]) Given a timed word  $w^t = (w(0), \tau(0))(w(1), \tau(1)) \dots$ , an MITL formula  $\varphi$  and a position  $i$  in the timed word, the satisfaction relation  $(w^t, i) \models \varphi$ , for  $i \geq 0$  (read  $w^t$  satisfies  $\varphi$  at position  $i$ ) is inductively defined as follows:

$$\begin{aligned} (w^t, i) \models p &\Leftrightarrow p \in w(i), \\ (w^t, i) \models \neg\varphi &\Leftrightarrow (w^t, i) \not\models \varphi, \\ (w^t, i) \models \varphi_1 \wedge \varphi_2 &\Leftrightarrow (w^t, i) \models \varphi_1 \text{ and } (w^t, i) \models \varphi_2, \\ (w^t, i) \models \bigcirc_I \varphi &\Leftrightarrow (w^t, i+1) \models \varphi \text{ and } \tau(i+1) - \tau(i) \in I, \\ (w^t, i) \models \diamond_I \varphi &\Leftrightarrow \exists j \geq i, \text{ such that } (w^t, j) \models \varphi, \tau(j) - \tau(i) \in I, \\ (w^t, i) \models \square_I \varphi &\Leftrightarrow \forall j \geq i, \tau(j) - \tau(i) \in I \Rightarrow (w^t, j) \models \varphi, \\ (w^t, i) \models \varphi_1 \mathcal{U}_I \varphi_2 &\Leftrightarrow \exists j \geq i, \text{ s.t. } (w^t, j) \models \varphi_2, \tau(j) - \tau(i) \in I \\ &\text{and } (w^t, k) \models \varphi_1, \forall i \leq k < j. \end{aligned}$$

We say that a timed run  $r^t = (r(0), \tau(0))(r(1), \tau(1)) \dots$  satisfies the MITL formula  $\varphi$  (we write  $r^t \models \varphi$ ) if and only if the corresponding timed word  $w(r^t) = (w(0), \tau(0))(w(1), \tau(1)) \dots$  with  $w(j) = L(r(j))$ ,  $\forall j \geq 0$ , satisfies the MITL formula  $(w(r^t) \models \varphi)$ .

It has been proved that MITL is decidable in infinite words and point-wise semantics, which is the case considered here (see [22, 23] for details). The model checking and satisfiability problems are *EXSPACE*-complete. It should be noted that in the context of timed systems, EXSPACE complexity is fairly low [24].

**Example 1.** Consider the WTS with  $S = \{s_0, s_1, s_2\}$ ,  $S_0 = \{s_0\}$ ,  $Act = \emptyset$ ,  $\rightarrow = \{(s_0, \emptyset, s_1), (s_1, \emptyset, s_2), (s_1, \emptyset, s_0), (s_2, \emptyset, s_1)\}$ ,  $d((s_0, \emptyset, s_1)) = 1.0$ ,  $d((s_1, \emptyset, s_2)) = 1.5$ ,  $d((s_1, \emptyset, s_0)) = 2.0$ ,  $d((s_2, \emptyset, s_1)) = 0.5$ ,  $AP = \{green\}$ ,  $L(s_0) = \{green\}$ ,  $L(s_1) = L(s_2) = \emptyset$  depicted in Figure 1.

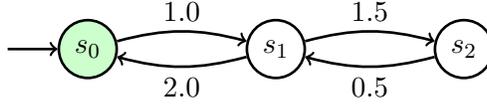


Figure 1.: An example of a WTS

Let two timed runs of the system be:

$$\begin{aligned} r_1^t &= (s_0, 0.0)(s_1, 1.0)(s_0, 3.0)(s_1, 4.0)(s_0, 6.0) \dots, \\ r_2^t &= (s_0, 0.0)(s_1, 1.0)(s_2, 2.5)(s_1, 3.0)(s_0, 5.0) \dots, \end{aligned}$$

and two MITL formulas  $\varphi_1 = \diamond_{[2,5]\{green\}}$ ,  $\varphi_2 = \square_{[0,5]\{green\}}$ . According to the MITL semantics, it follows that the timed run  $r_1^t$  satisfies  $\varphi_1$  ( $r_1^t \models \varphi_1$ ), since at the time stamp  $3.0 \in [2, 5]$  we have that  $L(s_0) = \{green\}$  so the atomic proposition *green* occurs at least once in the given interval. On the other hand, the timed run  $r_2^t$  does not satisfy  $\varphi_2$  ( $r_2^t \not\models \varphi_2$ ) since the atomic proposition *green* does not hold at every time stamp of the run  $r_2^t$  (it holds only at the time stamp 0.0).

## 2.6. Timed Büchi Automata

*Timed Büchi Automata (TBA)* were originally introduced in [19]. In this work, we partially adopt the notation from [24, 25]. Let  $C = \{c_1, \dots, c_{|C|}\}$  be a finite set of *clocks*. The set of *clock constraints*  $\Phi(C)$  is defined by the grammar:

$$\phi := \top \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid c \bowtie \psi,$$

where  $c \in C$  is a clock,  $\psi \in \mathbb{T}$  is a clock constant and  $\bowtie \in \{<, >, \geq, \leq, =\}$ . A *clock valuation* is a function  $\nu : C \rightarrow \mathbb{T}$  that assigns a value to each clock. A clock  $c_i$  has valuation  $\nu_i$  for  $i \in \{1, \dots, |C|\}$ , and  $\nu = (\nu_1, \dots, \nu_{|C|})$ . We denote by  $\nu \models \phi$  the fact that the valuation  $\nu$  satisfies the clock constraint  $\phi$ .

**Definition 7.** A *Timed Büchi Automaton* is a tuple  $\mathcal{A} = (Q, Q^{\text{init}}, C, \text{Inv}, E, F, AP, \mathcal{L})$  where  $Q$  is a finite set of locations;  $Q^{\text{init}} \subseteq Q$  is the set of initial locations;  $C$  is a finite set of clocks;  $\text{Inv} : Q \rightarrow \Phi(C)$  is the invariant;  $E \subseteq Q \times \Phi(C) \times 2^C \times Q$  gives the set of edges of the form  $e = (q, \gamma, R, q')$ , where  $q, q'$  are the source and target states,  $\gamma$  is the guard of edge  $e$  and  $R$  is a set of clocks to be reset upon executing the edge;  $F \subseteq Q$  is a set of accepting locations;  $AP$  is a finite set of atomic propositions; and  $\mathcal{L} : Q \rightarrow 2^{AP}$  labels every state with a subset of atomic propositions.

A state of  $\mathcal{A}$  is a pair  $(q, \nu)$  where  $q \in Q$  and  $\nu$  satisfies the *invariant*  $\text{Inv}(q)$ , i.e.,  $\nu \models \text{Inv}(q)$ . The initial state of  $\mathcal{A}$  is  $(q(0), (0, \dots, 0))$ , where  $q(0) \in Q^{\text{init}}$ . Given two states  $(q, \nu)$  and  $(q', \nu')$  and an edge  $e = (q, \gamma, R, q')$ , there exists a *discrete transition*  $(q, \nu) \xrightarrow{e} (q', \nu')$  iff  $\nu \models \gamma$ ,  $\nu' \models \text{Inv}(q')$ , and  $R$  is the *reset set*, i.e.,  $\nu'_i = 0$  for  $c_i \in R$  and  $\nu'_i = \nu_i$  for  $c_i \notin R$ . Given a  $\delta \in \mathbb{T}$ , there exists a *time transition*  $(q, \nu) \xrightarrow{\delta} (q', \nu')$  iff  $q = q', \nu' = \nu + \delta$  ( $\delta$  is summed component-wise) and  $\nu' \models \text{Inv}(q)$ . We write  $(q, \nu) \xrightarrow{\delta} \xrightarrow{e} (q', \nu')$  if there exists  $q'', \nu''$  such that  $(q, \nu) \xrightarrow{\delta} (q'', \nu'')$  and  $(q'', \nu'') \xrightarrow{e} (q', \nu')$  with  $q'' = q$ .

An infinite run of  $\mathcal{A}$  starting at state  $(q(0), \nu)$  is an infinite sequence of time and

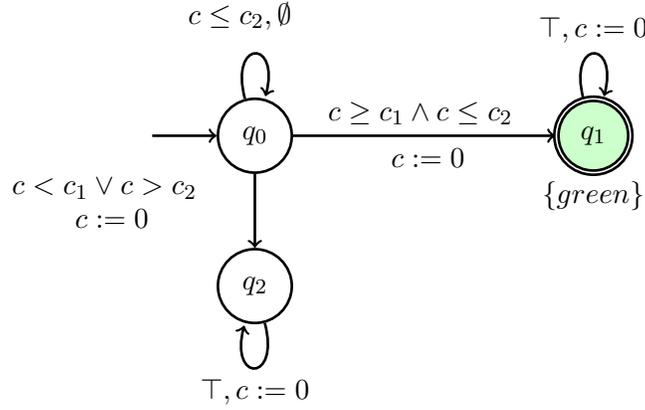


Figure 2.: A TBA  $\mathcal{A}$  that accepts the runs that satisfy formula  $\varphi = \diamond_{[c_1, c_2]} \{green\}$ .

discrete transitions  $(q(0), \nu(0)) \xrightarrow{\delta_0} (q(0)', \nu(0)') \xrightarrow{e_0} (q(1), \nu(1)) \xrightarrow{\delta_1} (q(1)', \nu(1)') \dots$ , where  $(q(0), \nu(0))$  is an initial state.

This run produces the timed word  $w = (\mathcal{L}(q(0)), \tau(0))(\mathcal{L}(q(1)), \tau(1)) \dots$  with  $\tau(0) = 0$  and  $\tau(i+1) = \tau(i) + \delta_i, \forall i \geq 1$ . The run is called *accepting* if  $q(i) \in F$  for infinitely many times. A timed word is *accepted* if there exists an accepting run that produces it. The problem of deciding the language emptiness of a given TBA is PSPACE-complete [19]. In other words, an accepting run of a given TBA can be synthesized, if one exists. In other words, an accepting run of a given TBA can be synthesized, if one exists. Any MITL formula  $\varphi$  over  $AP$  can be algorithmically translated into a TBA with the alphabet  $2^{AP}$ , such that the language of timed words that satisfy  $\varphi$  is the language of timed words produced by the TBA ([22, 26–28]).

**Example 2.** A TBA with  $Q = \{q_0, q_1, q_2\}, Q^{\text{init}} = \{q_0\}, C = \{c\}, \text{Inv}(q_0) = \text{Inv}(q_1) = \text{Inv}(q_2) = \emptyset, E = \{(q_0, \{c \leq c_2\}, \emptyset, q_0), (q_0, \{c \leq c_1 \vee c > c_2\}, c, q_2), (q_0, \{c \geq c_1 \wedge c \leq c_2\}, c, q_1), (q_1, \top, c, q_1), (q_2, \top, c, q_2)\}, F = \{q_1\}, AP = \{green\}, \mathcal{L}(q_0) = \mathcal{L}(q_2) = \emptyset, \mathcal{L}(q_1) = \{green\}$  that accepts all the timed words that satisfy the formula  $\varphi_3 = \diamond_{[c_1, c_2]} \{green\}$  is depicted in Figure 2. This formula will be used as reference for the following examples and simulations.

An example of a timed run of this TBA is  $(q_0, 0) \xrightarrow{\delta=\alpha_1} (q_0, \alpha_1) \xrightarrow{e=(q_0, \{c \geq c_1 \wedge c \leq c_2\}, c, q_1)} (q_1, 0) \dots$  with  $c_1 \leq \alpha_1 \leq c_2$ , which generates the timed word  $w^t = (\mathcal{L}(q_0), 0)(\mathcal{L}(q_0), \alpha_1)(\mathcal{L}(q_1), \alpha_1) \dots = (\emptyset, 0)(\emptyset, \alpha_1)(\{green\}, \alpha_1) \dots$  that satisfies the formula  $\varphi_3$ . The timed run  $(q_0, 0) \xrightarrow{\delta=\alpha_2} (q_0, \alpha_2) \xrightarrow{e=(q_0, \{c \leq c_1 \vee c > c_2\}, c, q_2)} (q_2, 0) \dots$  with  $\alpha_2 < c_1$ , generates the timed word  $w^t = (\mathcal{L}(q_0), 0)(\mathcal{L}(q_0), \alpha_2)(\mathcal{L}(q_2), \alpha_2) \dots = (\emptyset, 0)(\emptyset, \alpha_2)(\emptyset, \alpha_2) \dots$  that does not satisfy the formula  $\varphi_3$ .

**Remark 1.** Traditionally, the clock constraints and the TBAs are defined with  $\mathbb{T} = \mathbb{N}$ . However, they can be extended to accommodate  $\mathbb{T} = \mathbb{Q}_+$ , by multiplying all the rational numbers that are appearing in the state invariants and the edge constraints with their least common multiple.

### 3. Problem Formulation

#### 3.1. System Model

We focus on multi-agent systems with coupled dynamics of the form:

$$\dot{x}_i = - \sum_{j \in \mathcal{N}(i)} (x_i - x_j) + v_i, x_i \in \mathbb{R}^n, i \in \mathcal{I}. \quad (2)$$

The dynamics (2) consists of two parts; the first part is a consensus protocol representing the coupling between the agent and its neighbors, and the second one is a control input which will be exploited for high-level planning and is called free input. In this work, it is assumed that the free inputs are bounded by a positive constant  $v_{\max}$ , i.e.,  $\|v_i(t)\| \leq v_{\max}, \forall i \in \mathcal{I}, t \geq 0$ .

The topology of the multi-agent network is modeled through an undirected graph  $\mathcal{G} = (\mathcal{I}, \mathcal{E})$ , where  $\mathcal{I} = \{1, \dots, N\}$  and the following assumption is made.

**Assumption 1.** The communication graph  $\mathcal{G} = (\mathcal{I}, \mathcal{E})$  of the system is undirected, connected and static i.e., every agent preserves the same neighbors for all times.

#### 3.2. Specification

Our goal is to control the multi-agent system (2) so that each agent's behavior obeys a desired individual specification  $\varphi_i$  given in MITL. In particular, it is required to drive each agent to a desired subset of the *workspace*  $\mathbb{R}^n$  within certain time limits and provide certain atomic tasks there. Atomic tasks are captured through a finite set of *services*  $\Sigma_i, i \in \mathcal{I}$ . The position  $x_i$  of each agent  $i \in \mathcal{I}$  is labeled with services that are offered there. Thus, a *service labeling function*:

$$\Lambda_i : \mathbb{R}^n \rightarrow 2^{\Sigma_i}, \quad (3)$$

is introduced for each agent  $i \in \mathcal{I}$  which maps each state  $x_i \in \mathbb{R}^n$  to the subset of services  $\Lambda_i(x_i)$  which hold true at  $x_i$  i.e., the subset of services that the agent  $i$  can provide in position  $x_i$ . It is noted that although the term service labeling function it is used, these functions are not necessarily related to the labeling functions of a WTS as in Definition 4. Define also by  $\Lambda(x) = \bigcup_{i \in \mathcal{I}} \Lambda_i(x)$  the union of all the service labeling functions. We also assume that  $\Sigma_i \cap \Sigma_j = \emptyset$ , for all  $i, j \in \mathcal{I}, i \neq j$  which means that the agents do not share any services. Let us now introduce the following assumption which is necessary for formally defining the problem.

**Assumption 2.** There exists a decomposition  $S = \{S_\ell\}_{\ell \in \mathbb{I}}$  of the workspace  $\mathbb{R}^n$  which forms a cell decomposition according to Def. 1 and respects the labeling function  $\Lambda$  i.e., for all  $S_\ell \in S$  it holds that  $\Lambda(x) = \Lambda(x'), \forall x, x' \in S_\ell$ . This assumption implies that the same services hold at all the points that belong to the same cell of the decomposition.

Define for each agent  $i \in \mathcal{I}$  the labeling function:

$$\mathcal{L}_i : S \rightarrow 2^{\Sigma_i}, \quad (4)$$

which denotes the fact that when agent  $i$  visits a region  $S_\ell \in S$  and it can choose to *provide* a subset of the services that are being offered there i.e. it chooses to satisfy a

subset of  $\mathcal{L}_i(S_\ell)$ .

The trajectory of each agent  $i$  is denoted by  $x_i(t), t \geq 0, i \in \mathcal{I}$ . The trajectory  $x_i(t)$  is associated with a unique sequence  $r_{x_i}^t = (r_i(0), \tau_i(0))(r_i(1), \tau_i(1))(r_i(2), \tau_i(2)) \dots$ , of regions that the agent  $i$  crosses, where for all  $j \geq 0$  it holds that:  $x_i(\tau_i(j)) \in r_i(j)$  and  $\Lambda_i(x_i(t)) = \mathcal{L}_i(r_i(j)), \forall t \in [\tau_i(j), \tau_i(j+1))$  for some  $r_i(j) \in S$  and  $r_i(j) \neq r_i(j+1)$ . The timed word  $w_{x_i}^t = (\mathcal{L}_i(r_i(0)), \tau_i(0))(\mathcal{L}_i(r_i(1)), \tau_i(1))(\mathcal{L}_i(r_i(2)), \tau_i(2)) \dots$ , where  $w_i(j) = \mathcal{L}_i(r_i(j)), j \geq 0, i \in \mathcal{I}$ , is associated uniquely with the trajectory  $x_i(t)$ , and represents the sequence of services that *can be provided* by the agent  $i$  following the trajectory  $x_i(t), t \geq 0$ .

Define a *timed service word* as:

$$\tilde{w}_{x_i}^t = (\beta_i(z_0), \tilde{\tau}_i(z_0))(\beta_i(z_1), \tilde{\tau}_i(z_1))(\beta_i(z_2), \tilde{\tau}_i(z_2)) \dots, \quad (5)$$

where  $z_0 = 0 < z_1 < z_2 < \dots$  is a sequence of integers, and for all  $j \geq 0$  it holds that  $\beta_i(z_j) \subseteq \mathcal{L}_i(r_i(z_j))$  and  $\tilde{\tau}_i(z_j) \in [\tau_i(z_j), \tau_i(z_j + 1))$ . The timed service word is a sequence of services that are actually provided by agent  $i$  and is compliant with the trajectory  $x_i(t), t \geq 0$  by construction.

The specification task  $\varphi_i$  given as an MITL formula over the set of services  $\Sigma_i$  as in Def. 6, captures requirements on the services to be provided by agent  $i$ , for each  $i \in \mathcal{I}$ . We say that a trajectory  $x_i(t)$  satisfies a formula  $\varphi_i$  given in MITL over the set  $\Sigma_i$ , and formally write  $x_i(t) \models \varphi_i, \forall t \geq 0$ , if and only if there exists a *timed service word*  $\tilde{w}_{x_i}^t$  that complies with  $x_i(t)$  and satisfies  $\varphi_i$  according to the semantics of Def. 6.

**Example 3.** Consider  $N = 2$  agents performing in the partitioned environment of Figure 3. Both agents have the ability to pick up, deliver and throw two different balls. Their sets of services are  $\Sigma_1 = \{\text{pickUp1}, \text{deliver1}, \text{throw1}\}$  and  $\Sigma_2 = \{\text{pickUp2}, \text{deliver2}, \text{throw2}\}$ , respectively, and satisfy  $\Sigma_1 \cap \Sigma_2 = \emptyset$ . Three points of the agents' trajectories that belong to different cells with different services are captured. Assume that  $t_1 < t'_1 < t_2 < t'_2 < t_3 < t'_3$ . The trajectories  $x_1(t), x_2(t), t \geq 0$  are depicted with the red lines. According to Assumption 2, the cell decomposition  $S = \{S_\ell\}_{\ell \in \mathbb{I}} = \{S_1, \dots, S_6\}$  is given where  $\mathbb{I} = \{1, \dots, 6\}$  respects the labeling functions  $\Lambda_i, \mathcal{L}_i, i \in \{1, 2\}$ . In particular, it holds that:

$$\begin{aligned} \Lambda_1(x_1(t)) &= \mathcal{L}_1(r_1(0)) = \{\text{pickUp1}\}, t \in [0, t_1), \\ \Lambda_1(x_1(t)) &= \mathcal{L}_1(r_1(1)) = \{\text{throw1}\}, t \in [t_1, t_2), \\ \Lambda_1(x_1(t)) &= \mathcal{L}_1(r_1(2)) = \{\text{deliver1}\}, t \in [t_2, t_3), \\ \Lambda_1(x_1(t)) &= \mathcal{L}_1(r_1(3)) = \emptyset, t \geq t_3. \\ \Lambda_2(x_2(t)) &= \mathcal{L}_2(r_2(0)) = \{\text{pickUp2}\}, t \in [0, t'_1), \\ \Lambda_2(x_2(t)) &= \mathcal{L}_2(r_2(1)) = \{\text{deliver2}\}, t \in [t'_1, t'_2), \\ \Lambda_2(x_2(t)) &= \mathcal{L}_2(r_2(2)) = \{\text{throw2}\}, t \in [t'_2, t'_3), \\ \Lambda_2(x_2(t)) &= \mathcal{L}_2(r_2(3)) = \emptyset, t \geq t'_3. \end{aligned}$$

By the fact that  $w_i(j) = \mathcal{L}(r_i(j)), \forall i \in \{1, 2\}, j \in \{1, 2, 3\}$ , the corresponding individual timed words are given as:

$$\begin{aligned} w_{x_1}^t &= (\{\text{pickUp1}\}, 0)(\{\text{throw1}\}, t_1)(\{\text{deliver1}\}, t_2)(\emptyset, t_3), \\ w_{x_2}^t &= (\{\text{pickUp2}\}, 0)(\{\text{deliver2}\}, t'_1)(\{\text{throw2}\}, t'_2)(\emptyset, t'_3). \end{aligned}$$

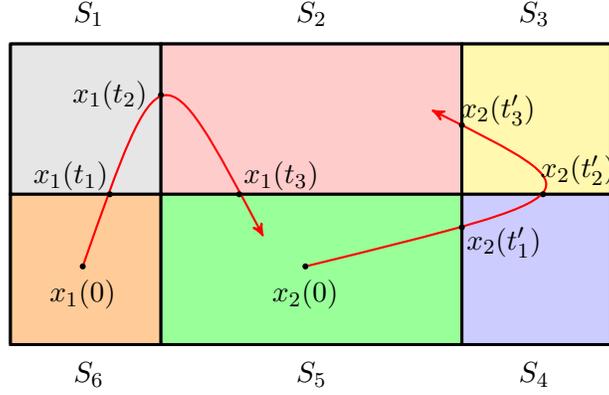


Figure 3.: An example of two agents performing in a partitioned workspace.

According to (5), two time service words (depicted with in Figure 3) are given as:

$$\begin{aligned}\tilde{w}_1^t &= (\beta_1(z_0), \tilde{\tau}_1(z_0))(\beta_1(z_1), \tilde{\tau}_1(z_1)), \\ \tilde{w}_2^t &= (\beta_2(z'_0), \tilde{\tau}_2(z'_0))(\beta_2(z'_1), \tilde{\tau}_2(z'_1)),\end{aligned}$$

where for agent 1 we have:  $z_0 = 0, z_1 = 2, \beta_1(z_0) = \{\text{pickUp1}\} \subseteq \mathcal{L}_1(r_1(z_0)), \beta_1(z_1) = \{\text{deliver1}\} \subseteq \mathcal{L}_1(r_1(z_1))$ . The corresponding elements for agent 2 are  $z'_0 = 0, z'_1 = 2, \beta_2(z'_0) = \{\text{pickUp2}\} \subseteq \mathcal{L}_2(r_2(z'_0)), \beta_2(z'_1) = \{\text{throw2}\} \subseteq \mathcal{L}_2(r_2(z'_1))$ . The time stamps  $\tilde{\tau}_1(z_0), \tilde{\tau}_1(z_1)$  should satisfy the following conditions:

$$\begin{aligned}\tilde{\tau}_1(z_0) &\in [\tau_1(z_0), \tau_1(z_0 + 1)) = [0, t_1), \\ \tilde{\tau}_1(z_1) &\in [\tau_1(z_1), \tau_1(z_1 + 1)) = [t_2, t_3), \\ \tilde{\tau}_2(z'_0) &\in [\tau_2(z'_0), \tau_2(z'_1)) = [0, t'_1), \\ \tilde{\tau}_2(z'_1) &\in [\tau_1(z'_1), \tau_1(z'_1 + 1)) = [t'_2, t'_3).\end{aligned}$$

### 3.3. Problem Statement

We are now ready to define the problem treated in this paper formally as follows:

**Problem 1.** *Given  $N$  agents that are governed by dynamics as in (2), modeled by the communication graph  $\mathcal{G}$ ,  $N$  task specification formulas  $\varphi_1, \dots, \varphi_N$  expressed in MITL over the sets of services  $\Sigma_1, \dots, \Sigma_N$ , respectively, service labeling functions  $\Lambda_1, \dots, \Lambda_N$ , as in (3), a cell decomposition  $S = \{S_\ell\}_{\ell \in \mathbb{I}}$  as in Assumption 2 and the labeling functions  $\mathcal{L}_1, \dots, \mathcal{L}_N$  given by (4), assign control laws to the free inputs  $v_1, \dots, v_N$  such that each agent fulfills its individual specification i.e.,  $x_i(t) \models \varphi_i, \forall i \in \mathcal{I}, t \geq 0$ , given the upper bound  $v_{max}$ .*

It should be noted that, in this work, the dependencies between the agents are induced through the coupled dynamics (2) and not in the discrete level, by allowing for couplings between the services (i.e.,  $\Sigma_i \cap \Sigma_j \neq \emptyset$ , for some  $i, j \in \mathcal{I}$ ). Hence, even though the agents do not share atomic propositions, the constraints on their motion due to the dynamic couplings may restrict them to fulfill the desired high-level tasks.

Treating additional couplings through individual atomic propositions in the discrete level, constitutes a far from trivial problem, which is a topic of current work.

**Remark 2.** In our previous work on the multi-agent controller synthesis framework under MITL specifications [10], the multi-agent system was considered to have fully-actuated dynamics. The only constraints on the system were due to the presence of time constrained MITL formulas. In the current framework, we have two types of constraints: the constraints due to the coupling dynamics of the system (2), which constrain the motion of each agent, and, the timed constraints that are inherently imposed from the time bounds of the MITL formulas. Thus, there exist formulas that cannot be satisfied either due to the coupling constraints or the time constraints of the MITL formulas. These constraints, make the procedure of the controller synthesis in the discrete level substantially different and more elaborate than the corresponding multi-agent LTL frameworks in the literature ([2, 29–31]).

**Remark 3.** The motivation for introducing the cell decomposition  $S = \{S_\ell\}_{\ell \in \mathbb{I}}$  in this Section, comes from the requirement to know a priori which services hold in each part of the workspace. As will be clarified through the problem solution, this is necessary since the abstraction of the workspace (which is part of our proposed solution) may not be compliant with the initial given cell decomposition, and thus, new cell decompositions might be required.

## 4. Proposed Solution

In this section, a systematic solution to Problem 1 is introduced. Our overall approach builds on abstracting system (2) through a WTS for each agent and exploiting the fact that the timed runs in the  $i$ -th WTS project onto the trajectories of agent  $i$  while preserving the satisfaction of the individual MITL formulas  $\varphi_i, i \in \mathcal{I}$ . The following analysis is performed:

- (1) Initially, the boundedness of the agents’ relative positions is proved, in order to guarantee boundedness of the coupling terms  $-\sum_{j \in \mathcal{N}(i)}(x_i - x_j)$ . This property is required for the derivation of the symbolic models. (Section 4.1).
- (2) We utilize decentralized abstraction techniques for the multi-agent system, i.e., a discretization of both the workspace and time in order to model the motion capabilities of each agent by a WTS  $\mathcal{T}_i, i \in \mathcal{I}$  (Section 4.2).
- (3) Given the WTSs, consistent runs are defined in order to take into consideration the coupling constraints among the agents. The computation of the product of the individual WTSs is also required (Section 4.3).
- (4) A five-step automated procedure for controller synthesis which serves as a solution to Problem 1 is provided in Section 4.4.
- (5) Finally, the computational complexity of the proposed approach is discussed in Section 4.5.

The next sections provide the proposed solution in detail.

### 4.1. Boundedness Analysis

**Theorem 1.** *Consider the multi-agent system (2) modeled by the undirected communication graph  $\mathcal{G}$ . Assume that the network graph is connected (i.e.  $\lambda_2(\mathcal{G}) > 0$ ) and*

let  $v_i, i \in \mathcal{I}$  satisfy  $\|v_i(t)\| \leq v_{max}, \forall i \in \mathcal{I}, t \geq 0$ . Furthermore, let  $\bar{R} > K_2 v_{max}$  be a positive constant, where  $K_2 = \frac{2\sqrt{N(N-1)}\|D(\mathcal{G})^\top\|}{\lambda_2^2(\mathcal{G})} > 0$  and where  $D(\mathcal{G})$  is the network adjacency matrix. Then, for each initial condition  $x_i(0) \in \mathbb{R}^n$ , there exists a time  $T > 0$  such that  $\tilde{x}(t) \in \mathcal{X}, \forall t \geq T$ , where  $\mathcal{X} = \{x \in \mathbb{R}^{Nn} : \|\tilde{x}\| \leq \bar{R}\}$  and with  $\tilde{x}(t)$  as given in (1).

**Proof.** The proof can be found in Appendix A. □

It should be noticed that the relative boundedness of the agents' positions guarantees a global bound on the coupling terms  $-\sum_{j \in \mathcal{N}(i)}(x_i - x_j)$ , as defined in (2). This bound will be later exploited in order to capture the behavior of the system in  $\mathcal{X} = \{x \in \mathbb{R}^{Nn} : \|\tilde{x}\| \leq \bar{R}\}$ , by a discrete state WTS.

## 4.2. Abstraction

In this section we provide the abstraction technique that is adopted from our previous work [16] in order to capture the dynamics of each agent into Transition Systems. Thereafter, we work completely in discrete level, which is necessary in order to solve Problem 1.

Firstly, some additional notation is introduced. Given an index set  $\mathbb{I}$  and an agent  $i \in \mathcal{I}$  with neighbors  $j_1, \dots, j_{N_i} \in \mathcal{N}(i)$ , define the mapping  $\text{pr}_i : \mathbb{I}^N \rightarrow \mathbb{I}^{N_i+1}$  which assigns to each  $N$ -tuple  $\mathbf{l} = (l_1, \dots, l_N) \in \mathbb{I}^N$  the  $N_i + 1$  tuple  $\mathbf{l}_i = (l_i, l_{j_1}, \dots, l_{j_{N_i}}) \in \mathbb{I}^{N_i+1}$  which denotes the indices of the cells where the agent  $i$  and its neighbors belong.

### 4.2.1. Well-Posed Abstractions

Loosely speaking, an abstraction is characterized by a discretization of the workspace into cells, which we denote by  $\bar{S} = \{\bar{S}_l\}_{l \in \bar{\mathbb{I}}}$ , a time step  $\delta t$  and selection of feedback laws in place of the free inputs  $v_i(t), \forall i \in \mathcal{I}$ . The time step  $\delta t$  models the time that an agent needs to transit from one cell to another, and  $v_i(t)$  is the controller that guarantees such a transition. Note that the time step  $\delta t$  is the same for all the agents. Let us denote by  $(\bar{S}, \delta t)$  the aforementioned *space-time discretization*.

Before defining formally the concept of well-posed abstractions, an intuitive graphical representation is provided. Consider a cell decomposition  $\bar{S} = \{\bar{S}_l\}_{l \in \bar{\mathbb{I}} = \{1, \dots, 12\}}$  as depicted in Figure 4 and a time step  $\delta t$ . The tails and the tips of the arrows in the figure depict the initial cell and the endpoints of agent's  $i$  trajectories at time  $\delta t$  respectively. In both cases in the figure we focus on agent  $i$  and consider the same cell configuration for  $i$  and its neighbors. By configuration we mean the cell that the agent  $i$  and its neighbors belong at a current time. However, different dynamics are considered for Cases (i) and (ii). In Case (i), it can be observed that for the three distinct initial positions in cell  $\bar{S}_{l_i}$ , it is possible to drive agent  $i$  to cell  $\bar{S}_{l'_i}$  at time  $\delta t$ . We assume that this is possible for all initial conditions in this cell and irrespectively of the initial conditions of  $i$ 's neighbors in their cells and the inputs they choose. It is also assumed that this property holds for all possible cell configurations of  $i$  and for all the agents of the system. Thus, we have a *well-posed discretization* for system (i). On the other hand, for the same cell configuration and system (ii), the following can be observed. For three distinct initial conditions of  $i$  the corresponding reachable sets at  $\delta t$ , which are enclosed in the dashed circles, lie in different cells. Thus, it is not possible given this cell configuration of  $i$  to find a cell in the decomposition which is

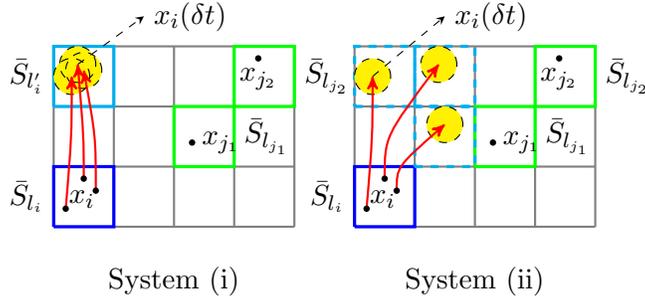


Figure 4.: Illustration of a space-time discretization which is well posed for system (i) but non-well posed for system (ii).

reachable from every point in the initial cell and we conclude that discretization is not well-posed for system (ii).

More specifically, consider a  $(\bar{S}, \delta t)$ -space-time discretization which is the outcome of the abstraction technique that is designed for the problem solution and will be presented in Section 4.2.3. Let  $\bar{S} = \{\bar{S}_l\}_{l \in \mathbb{I}}$  be a cell decomposition in which the agent  $i$  occupies the cell  $\bar{S}_{l_i}$ ,  $\delta t$  be a time step and  $\bar{d}_{\max} = \sup\{\|x - y\| : x, y \in \bar{S}_l, l \in \mathbb{I}\}$  be the diameter of the cell decomposition  $\bar{S}$ . It should be noted that this decomposition is not necessarily the same cell decomposition  $S$  from Assumption 2 and Problem 1. Through the aforementioned space and time discretization  $(\bar{S}, \delta t)$  we aim to capture the reachability properties of the continuous system (2), in order to create a WTS for each agent. If there exists a free input for each state in  $\bar{S}_{l_i}$  that navigates the agent  $i$  into the cell  $\bar{S}_{l'_i}$  precisely in time  $\delta t$ , regardless of the locations of the agent  $i$ 's neighbors within their current cells, then a transition from  $l_i$  to  $l'_i$  is enabled in the WTS. This forms the well-possessedness of transitions. A more detailed mathematical derivation as well as feedback laws  $v_i(t), i \in \mathcal{I}$  which guarantee a well-posed space-time discretization  $(\bar{S}, \delta t)$  can be found in our previous work [16].

#### 4.2.2. Sufficient Conditions

We present at this point the sufficient conditions that relate the dynamics of the multi-agent system (2), the time step  $\delta t$  and the diameter  $\bar{d}_{\max}$ , and guarantee the existence of the aforementioned well-posed transitions for each cell. Based on our previous work [16] (Section III, inequality (3), Section IV, inequalities (28, 29)), in order to derive well-posed abstractions, a nonlinear system of the form:

$$\dot{x}_i = f_i(x_i, \mathbf{x}_j) + v_i, i \in \mathcal{I}, \quad (6)$$

where  $\mathbf{x}_j = (x_{j_1}, \dots, x_{j_{N_i}}) \in \mathbb{R}^{N_i n}$ , should fulfill the following sufficient conditions:  
**(C1)** There exists  $M > v_{\max} > 0$  such that  $\|f_i(x_i, \mathbf{x}_j)\| \leq M, \forall i \in \mathcal{I}, \forall x \in \mathbb{R}^{N_i n} : \text{pr}_i(x) = (x_i, \mathbf{x}_j)$  and  $\tilde{x} \in \mathcal{X}$ , by applying the projection operator  $\text{pr}_i$  for  $\mathbb{I} = \mathbb{R}^n$ .  
**(C2)** There exists a Lipschitz constant  $L_1 > 0$  such that:

$$\|f_i(x_i, \mathbf{x}_j) - f_i(x_i, \mathbf{y}_j)\| \leq L_1 \|(x_i, \mathbf{x}_j) - (x_i, \mathbf{y}_j)\|, \forall i \in \mathcal{I}, x_i, y_i \in \mathbb{R}^n, \mathbf{x}_j, \mathbf{y}_j \in \mathbb{R}^{N_i n}. \quad (7)$$

**(C3)** There exists a Lipschitz constant  $L_2 > 0$  such that:

$$\|f_i(x_i, \mathbf{x}_j) - f_i(y_i, \mathbf{x}_j)\| \leq L_2 \|(x_i, \mathbf{x}_j) - (y_i, \mathbf{x}_j)\|, \forall i \in \mathcal{I}, x_i, y_i \in \mathbb{R}^n, \mathbf{x}_j, \mathbf{y}_j \in \mathbb{R}^{N_i n}.$$

From (2) and (6) we get  $f_i(x_i, \mathbf{x}_j) = - \sum_{j \in \mathcal{N}(i)} (x_i - x_j)$ . By checking all the conditions

one by one for  $f_i(x_i, \mathbf{x}_j)$  as in (2), it can be shown that our system satisfies all the conditions **(C1)**-**(C3)**. The proof can be found in Appendix C.

Based on the sufficient condition for well posed abstractions in [16], the diameter  $\bar{d}_{\max}$  and the time step  $\delta t$  of the discretization  $\bar{S}, \delta t$  can be selected as:

$$\bar{d}_{\max} \in \left( 0, \frac{(1-\lambda)^2 v_{\max}^2}{4ML} \right], \quad (8a)$$

$$\delta t \in \left[ \frac{(1-\lambda)v_{\max} - \sqrt{(1-\lambda)^2 v_{\max}^2 - 4ML\bar{d}_{\max}}}{2ML}, \frac{(1-\lambda)v_{\max} + \sqrt{(1-\lambda)^2 v_{\max}^2 - 4ML\bar{d}_{\max}}}{2ML} \right], \quad (8b)$$

where  $L = \max\{3L_2 + 4L_1\sqrt{N_i}, i \in \mathcal{I}\}$  and with the dynamics bound  $M$  and the Lipschitz constants  $L_1, L_2$  as previously defined. Furthermore,  $\lambda \in (0, 1)$  is a design parameter which quantifies the part of the free input that is additionally exploited for reachability purposes. In particular, given an agent's initial cell configuration, the agent can reach any point inside an appropriate ball at  $\delta t$  through a parameterized feedback law in place of the free input  $v_i$ . The radius of this ball increases proportionally to the value of  $\lambda$ , and thus, also the number of the agent's successor cells, which are the ones intersecting the ball. It is noted that an increasing choice of  $\lambda$  results in finer discretizations, therefore providing a quantifiable trade-off between the discrete model's accuracy and complexity. Furthermore, it follows from the acceptable values of  $\bar{d}_{\max}$  that the cells can be selected coarser, when (i) the available control  $v_{\max}$  is larger, and, (ii) the coupling term bound  $M$  together with the dynamics' variation, which is captured through the parameter  $L$ , are smaller. Analogous restrictions need to hold for the time step  $\delta t$ . In particular, the time step cannot be selected very large, because the required control for the manipulation of the coupling terms increases due to the evolution of the agent's neighbors during the transition interval. Finally, the time step cannot be selected very small either, because controlling the agent to the same point from each initial condition in its cell, will require a large control effort over a very short transition interval.

**Remark 4.** Assume that a cell-decomposition of diameter  $\bar{d}_{\max}$  and a time step  $\delta t$  which guarantee well-posed transitions, namely, which satisfy (8a) and (8b), have been chosen. Then, it is also possible to choose any other cell-decomposition with diameter  $\hat{d}_{\max} \leq \bar{d}_{\max}$  since, by (8b), the range of acceptable  $\delta t$  increases.

Having shown that the dynamics of system (2) satisfy the sufficient conditions **(C1)**-**(C3)**, a well-posed space-time discretization  $(\bar{S}, \delta t)$  has been obtained. Recall now Assumption 2. It remains to establish the compliance of the cell decomposition  $S = \{S_\ell\}_{\ell \in \mathbb{I}}$ , which is given in the statement of Problem 1, with the cell decomposition  $\bar{S} = \{\bar{S}_l\}_{l \in \bar{\mathbb{I}}}$ , which is the outcome of the abstraction. By the term of compliance, we

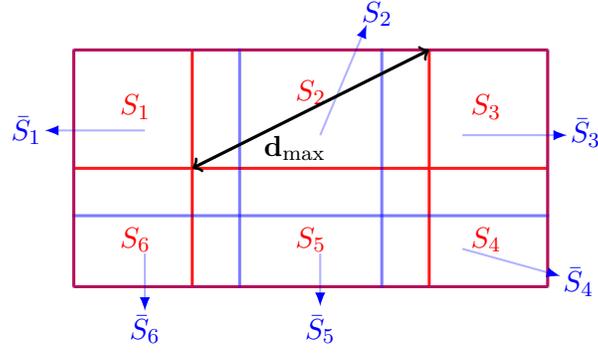


Figure 5.: An example with a given cell decomposition  $S = \{S_l\}_{l \in \{1, \dots, 6\}}$  from Problem 1 and a non-compliant cell decomposition  $\bar{S} = \{\bar{S}_l\}_{l \in \bar{\mathbb{I}} = \{1, \dots, 6\}}$  which is the outcome of the proposed abstraction technique.

mean that:

$$\bar{S}_l \cap S_\ell \in S \cup \{\emptyset\}, \forall \bar{S}_l \in \bar{S}, S_\ell \in S, l \in \bar{\mathbb{I}}, \ell \in \mathbb{I}.$$

In order to address this problem, define:

$$\hat{S} = \{\hat{S}_l\}_{l \in \hat{\mathbb{I}}} = \{\bar{S}_l \cap S_\ell : l \in \bar{\mathbb{I}}, \ell \in \mathbb{I}\} \setminus \{\emptyset\},$$

which forms a cell decomposition and is compliant with the cell decomposition  $S$  from Problem 1 with diameter  $\hat{d}_{\max} = \sup\{\|x - y\| : x, y \in \hat{S}_l, l \in \hat{\mathbb{I}}\} \leq \bar{d}_{\max}$  and serves as the abstraction solution of this problem. By taking all the intersections  $\bar{S}_l \cap S_\ell, \forall l \in \bar{\mathbb{I}}, \forall \ell \in \mathbb{I}$  and enumerating them through the index set  $\hat{\mathbb{I}}$ , the cells  $\{\hat{S}_l\}_{l \in \hat{\mathbb{I}}}$  are constructed. For the cells  $\{\hat{S}_l\}_{l \in \hat{\mathbb{I}}}$ , the following holds:  $\forall \ell \in \mathbb{I}, \exists l \in \bar{\mathbb{I}}$  such that  $\hat{S}_l = \bar{S}_l \cap S_\ell$  and  $\text{int}(\hat{S}_l) \cap \text{int}(\hat{S}_{l'}) \neq \emptyset$  for all  $l' \in \hat{\mathbb{I}} \setminus \{l\}$ . After all the intersections we have  $\cup_{l \in \hat{\mathbb{I}}} \hat{S}_l = \mathcal{X}$ . The diameter of the cell decomposition  $\hat{S} = \{\hat{S}_l\}_{l \in \hat{\mathbb{I}}}$  is defined as  $\hat{d}_{\max} = \sup\{\|x - y\| : x, y \in \hat{S}_l, l \in \hat{\mathbb{I}}\} \leq \bar{d}_{\max}$ . Hence, according to Remark 4, we have a well-posed abstraction. The following Example is an illustration of these derivations.

**Example 4.** Let  $S = \{S_\ell\}_{\ell \in \{1, \dots, 6\}}$  be the cell decomposition of Problem 1, which is depicted in Figure 5 by the red rectangles. In the same figure, we illustrate the cell decomposition  $\bar{S} = \{\bar{S}_l\}_{l \in \bar{\mathbb{I}} = \{1, \dots, 6\}}$ .  $\bar{S}$  serves as potential solution of this Problem satisfying all the abstraction properties that have been mentioned in this Section. It can be observed that the two cell decompositions are not compliant according to (4.2.2). However, by using the methodology below Remark 4, a new cell decomposition  $\hat{S} = \{\hat{S}_l\}_{l \in \hat{\mathbb{I}} = \{1, \dots, 15\}}$ , which is compliant with  $S$  and has 15 regions, can be obtained.  $\hat{S}$  forms the final cell decomposition solution of Problem 1 and is depicted in Figure 6. Let also  $\bar{d}_{\max}, \hat{d}_{\max}$  be the diameters of the cell decompositions  $\bar{S}, \hat{S}$  respectively. Then, it holds that  $\hat{d}_{\max} \leq \bar{d}_{\max}$ , which is in accordance with Remark 4.

$\hat{S}_1$	$\hat{S}_2$	$\hat{S}_3$	$\hat{S}_4$	$\hat{S}_5$
$\hat{S}_{10}$	$\hat{S}_9$	$\hat{S}_8$	$\hat{S}_7$	$\hat{S}_6$
$\hat{S}_{11}$	$\hat{S}_{12}$	$\hat{S}_{13}$	$\hat{S}_{14}$	$\hat{S}_{15}$

Figure 6.: The resulting compliant cell decomposition  $\hat{S} = \{\hat{S}_i\}_{i \in \hat{\mathbb{I}} = \{1, \dots, 15\}}$  of the Example 4 which serves as solution to Problem 1.

#### 4.2.3. Discrete System Abstraction

For the solution to Problem 1, the WTS of this agent which corresponds to the cell decomposition  $\hat{S}$  with diameter  $\hat{d}_{\max}$  and the time step  $\delta t$  will be exploited. Thus, the WTS of each agent is defined as follows:

**Definition 8.** The motion of each agent  $i \in \mathcal{I}$  in the workspace is modeled by the WTS  $\mathcal{T}_i = (S_i, S_i^{\text{init}}, Act_i, \longrightarrow_i, d_i, AP_i, \hat{L}_i)$  where:

- $S_i = \hat{\mathbb{I}}$  is the set of states of each agent which is the set of indices of the cell decomposition.
- $S_i^{\text{init}} \subseteq S_i$  is a set of initial states defined by the agents' initial positions in the workspace.
- $Act_i = \hat{\mathbb{I}}^{N_i+1}$ , the set of actions representing where agent  $i$  and its neighbors are located.
- For a pair  $(l_i, \mathbf{l}_i, l'_i)$  we have that  $(l_i, \mathbf{l}_i, l'_i) \in \longrightarrow_i$  iff  $l_i \xrightarrow{\mathbf{l}_i} l'_i$  is well-posed for each  $l_i, l'_i \in S_i$  and  $\mathbf{l}_i = (l_{j_1}, \dots, l_{j_{N_i}}) \in Act_i$ .
- $d_i : \longrightarrow_i \rightarrow \mathbb{T}$ , is a map that assigns a positive weight (duration) to each transition. The duration of each transition is exactly equal to  $\delta t > 0$ .
- $AP_i = \Sigma_i$ , is the set of atomic propositions which are inherent properties of the workspace.
- $L_i : S_i \rightarrow 2^{AP_i}$ , is the labeling function that maps every state  $s \in S_i$  into the services that can be provided in this state.

The individual WTSs of the agents will allow us to work completely in the discrete level and design sequences of controllers that solve Problem 1.

Every WTS  $\mathcal{T}_i, i \in \mathcal{I}$  generates timed runs and timed words of the form  $r_i^t = (r_i(0), \tau_i(0))(r_i(1), \tau_i(1))(r_i(2), \tau_i(2)) \dots$ ,  $w_i^t = (L_i(r_i(0)), \tau_i(0))(L_i(r_i(1)), \tau_i(1))(L_i(r_i(2)), \tau_i(2)) \dots$  respectively, over the set  $2^{AP_i}$  according to Def. 5 with  $\tau_i(j) = j\delta t, \forall j \geq 0$ . The relation between the timed words that are generated by the WTSs  $\mathcal{T}_i, i \in \mathcal{I}$  with the timed service words produced by the trajectories  $x_i(t), i \in \mathcal{I}, t \geq 0$  is provided through the following remark:

**Remark 5.** By construction, each timed word produced by the WTS  $\mathcal{T}_i$  is a service timed word associated with the trajectory  $x_i(t)$  of the system (2). Hence, if we find a timed word of  $\mathcal{T}_i$  satisfying a formula  $\varphi_i$  given in MITL, we also find for each agent  $i$  a desired timed word of the original system, and hence trajectories  $x_i(t)$  that are a solution to the Problem 1. Therefore, the produced timed words of  $\mathcal{T}_i$  are compliant with the service timed words of the trajectories  $x_i(t)$ .

### 4.3. Runs Consistency

Due to the coupled dynamics between the agents, it is required that each individual agent's run is compliant with the corresponding discrete trajectories of its neighbors, which determine the actions in the agent's run. Therefore, even though we have the individual WTS of each agent, the runs that the latter generates may not be performed by an agent due to the constrained motion that is imposed by the coupling terms. Hence, we need to synchronize the agents at each time step  $\delta t$  and determine which of the generated runs of the individual WTS can be performed by the agent. Hereafter, they will be called *consistent runs*. In order to address the aforementioned issue, we provide a centralized product WTS which captures the behavior of the coupled multi-agent system as a team, and the generated product run (see Def. 10) can later be projected onto consistent individual runs. The following two definitions deal with the product WTS and consistent runs respectively.

**Definition 9.** Given the individual WTSs  $\mathcal{T}_i, i \in \mathcal{I}$  from Def. 8, the product WTS  $\mathcal{T}_p = (S_p, S_p^{\text{init}}, \longrightarrow_p, L_p)$  is defined as follows:

- $S_p = \hat{\mathbb{I}}^N$ ;
- $(s_1, \dots, s_N) \in S_p^{\text{init}}$  if  $s_i \in S_i^{\text{init}}, \forall i \in \mathcal{I}$ ;
- $(\mathbf{l}, \mathbf{l}') \in \longrightarrow_p$  iff  $l'_i \in \text{Post}_i(l_i, \text{pr}_i(\mathbf{l})), \forall i \in \mathcal{I}, \forall \mathbf{l} = (\mathbf{l}_1, \dots, \mathbf{l}_N), \mathbf{l}' = (\mathbf{l}'_1, \dots, \mathbf{l}'_N)$ ;
- $L_p : \hat{\mathbb{I}}^N \rightarrow 2^{\cup_{i=1}^N \Sigma_i}$  defined as  $L_p(\mathbf{l}) = \cup_{i=1}^N L_i(l_i)$ ;
- $d_p : \longrightarrow_p \rightarrow \mathbb{T}$ : as in the individual WTS's case, with transition weight  $d_p(\cdot) = \delta t$ .

**Definition 10.** Given a timed run:

$$r_p^t = ((r_p^1(0), \dots, r_p^N(0)), \tau_p(0))((r_p^1(1), \dots, r_p^N(1)), \tau_p(1)) \dots,$$

that is generated by the product WTS  $\mathcal{T}_p$ , the induced set of projected runs

$$\{r_i^t = (r_p^i(0), \tau_p(0))(r_p^i(1), \tau_p(1)) \dots : i \in \mathcal{I}\},$$

of the WTSs  $\mathcal{T}_1, \dots, \mathcal{T}_N$ , respectively will be called *consistent runs*. Since the duration of each agent's transition is  $\delta t$  it holds that  $\tau_p(j) = j\delta t, j \geq 0$ .

Therefore, through the product WTS  $\mathcal{T}_p$ , we can always generate individual consistent runs for each agent. It remains to provide a systematic approach of how to determine consistent runs  $\tilde{r}_1, \dots, \tilde{r}_N$  which are associated with the corresponding time serviced words  $\tilde{w}_1^t, \dots, \tilde{w}_N^t$ . Note that we use the tilde accent to denote timed runs and words that correspond to the problem solution. The corresponding compliant trajectories  $x_1(t), \dots, x_N(t)$  of the timed words  $\tilde{w}_1^t, \dots, \tilde{w}_N^t$  satisfy the corresponding MITL formulas  $\varphi_1, \dots, \varphi_N$ , and they are a solution to Problem 1. This follows from the fact that the product transition system is simulated by the  $\delta t$ -sampled version of the continuous system (see [32] for the definition of a simulation relation). In particular, let  $\mathcal{T}_{\delta t}$  be the  $\delta t$ -sampled WTS of system (1), as defined in [32, Def. 11.4], with labeling function  $L_{\delta t} : \mathbb{R}^{Nn} \rightarrow 2^{\cup_{i=1}^N \Sigma_i}$  given as  $L_{\delta t}(x_1, \dots, x_N) = \cup_{i=1}^N \Lambda_i(x_i)$  and  $\Lambda_i$  as defined in Section 2. Consider also the WTS  $\mathcal{T}_p$  and the relation  $\mathcal{R} \subseteq S_p \times \mathcal{X}^N$  given as  $(\mathbf{l}, (x_1, \dots, x_N)) \in \mathcal{R}$ , iff  $(x_1, \dots, x_N) \in S_{l_1} \times \dots \times S_{l_N}$ , where  $\mathbf{l} = (l_1, \dots, l_N)$ . Then, from the definition of the agent's individual transitions in each WTS  $\mathcal{T}_i$  and the fact that for all points in a cell the same atomic proposition hold true, it can be deduced that  $\mathcal{R}$  is a simulation relation from  $\mathcal{T}_p$  to the  $\delta t$ -sampled WTS  $\mathcal{T}_{\delta t}$ .

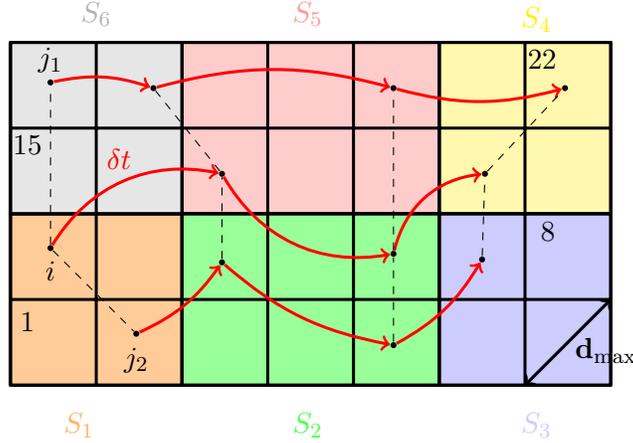


Figure 7.: Timed runs of the agents  $i, j_1, j_2$

**Remark 6.** We chose to utilize decentralized abstractions, to generate the individual WTSs  $\mathcal{I}, i \in \mathcal{I}$  for each agent and to compute the synchronized-centralized product WTS  $\mathcal{T}_p$  for the following reasons:

- (1) The state space of the centralized system to be abstracted is  $\mathcal{X}^N \subseteq \mathbb{R}^{Nn}$ , which is harder to visualize and handle as well as not naturally related to the individual specifications. Thus, it is more natural to define the specifications through the individual transition system of each agent corresponding to a discretization of  $\mathcal{X}$  and then generate the product in order to obtain potential consistent satisfying plans.
- (2) Additionally, many centralized abstraction frameworks are based on approximations of the system's reachable sets from a given cell over the transition time interval. These frameworks, require in the general nonlinear case, global dynamics properties and may avoid taking into account the finer dynamics properties of the individual entities, which can lead to more conservative estimates for large scale systems.

We provide here an example that explains the notation that has been introduced until now.

**Example 5.** Consider  $N = 3$  agents performing in the workspace with  $\mathcal{N}(i) = \{1, 2\}$  as depicted in Figure 7.  $S = \{S_\ell\}_{\ell \in \mathbb{I}=\{1, \dots, 6\}}$  is the given cell decomposition from Problem 1 and  $\bar{S} = \{\bar{S}_l\}_{l \in \mathbb{I}=\{1, \dots, 28\}}$  is the cell decomposition which is the outcome of the proposed abstraction technique. Let the atomic propositions be  $\{p_1, \dots, p_6\} = \{\text{orange}, \text{green}, \text{blue}, \text{yellow}, \text{red}, \text{grey}\}$ . The red arrows represent both the transitions of the agent  $i$  and its neighbors. The dashed lines indicate the edges in the network graph. For the atomic propositions we have that  $L_i(14) = \{p_1\}, L_i(17) = \{p_5\}, L_i(10) = \{p_2\}, L_i(20) = \{p_4\}, L_{j_1}(28) = \{p_6\} = L_{j_1}(27), L_{j_1}(24) = \{p_5\}, L_{j_1}(22) = \{p_4\}, L_{j_2}(2) = \{p_1\}, L_{j_2}(12) = \{p_2\} = L_{j_2}(5), L_{j_2}(9) = \{p_3\}$ . Note also the diameter of the cells  $\hat{d}_{\max} = \bar{d}_{\max}$ . For the cell

configurations we have:

$$\begin{array}{l}
\text{Init } (t = 0) : \begin{cases} \mathbf{l}_i = (14, 28, 2), \\ \mathbf{l}_{j_1} = (28, 14), \\ \mathbf{l}_{j_2} = (2, 14), \end{cases} \quad \text{Step 1 } (t = \delta t) : \begin{cases} \mathbf{l}_i = (17, 27, 13), \\ \mathbf{l}_{j_1} = (27, 17), \\ \mathbf{l}_{j_2} = (13, 17), \end{cases} \\
\text{Step 2 } (t = 2\delta t) : \begin{cases} \mathbf{l}_i = (10, 24, 5), \\ \mathbf{l}_{j_1} = (24, 10), \\ \mathbf{l}_{j_2} = (5, 10), \end{cases} \quad \text{Step 3 } (t = 3\delta t) : \begin{cases} \mathbf{l}_i = (20, 22, 9), \\ \mathbf{l}_{j_1} = (22, 20), \\ \mathbf{l}_{j_2} = (9, 20), \end{cases}
\end{array}$$

which are actions to the corresponding transitions. Three consistent timed runs are given as:

$$\begin{aligned}
r_i^t &= (r_i(0) = 14, \tau_i(0) = 0)(r_i(1) = 17, \tau_i(1) = \delta t)(r_i(2) = 10, \tau_i(2) = 2\delta t) \\
&\quad (r_i(3) = 20, \tau_i(3) = 3\delta t), \\
r_{j_1}^t &= (r_{j_1}(0) = 28, \tau_{j_1}(0) = 0)(r_{j_1}(1) = 27, \tau_{j_1}(1) = \delta t)(r_{j_1}(2) = 24, \tau_{j_1}(2) = 2\delta t) \\
&\quad (r_{j_1}(3) = 22, \tau_{j_1}(3) = 3\delta t), \\
r_{j_2}^t &= (r_{j_2}(0) = 2, \tau_{j_2}(0) = 0)(r_{j_2}(1) = 13, \tau_{j_2}(1) = \delta t)(r_{j_2}(2) = 5, \tau_{j_2}(2) = 2\delta t) \\
&\quad (r_{j_2}(3) = 9, \tau_{j_2}(3) = 3\delta t).
\end{aligned}$$

It can be observed that  $r_i^t \models (\varphi_i = \diamond_{[0,6]} \{\text{yellow}\})$  if  $3\delta t \in [0, 6]$ ,  $r_{j_1}^t \models (\varphi_{j_1} = \diamond_{[3,10]} \{\text{red}\})$  if  $2\delta t \in [3, 10]$  and  $r_{j_2}^t \models (\varphi_{j_2} = \diamond_{[3,9]} \{\text{blue}\})$  if  $3\delta t \in [3, 9]$ .

#### 4.4. Controller Synthesis

The proposed controller synthesis procedure is described with the following steps:

- (1)  $N$  TBAs  $\mathcal{A}_i$ ,  $i \in \mathcal{I}$  that accept all the timed runs satisfying the corresponding specification formulas  $\varphi_i$ ,  $i \in \mathcal{I}$  are constructed.
- (2) A Büchi WTS  $\tilde{\mathcal{T}}_i = \mathcal{T}_i \otimes \mathcal{A}_i$  (see Def. 11 below) for every  $i \in \mathcal{I}$  is constructed. The accepting runs of  $\tilde{\mathcal{T}}_i$ , computed using standard graph search algorithms, are the individual runs of the  $\mathcal{T}_i$  that satisfy the corresponding MITL formula  $\varphi_i$ ,  $i \in \mathcal{I}$ .
- (3) We pick a set of accepting runs  $\{\tilde{r}_1^t, \dots, \tilde{r}_N^t\}$  from Step 2. We check if they are consistent according to Def. 10. If this is true then we proceed with Step 5. If this is not true then we repeat Step 3 with a different set of accepting runs. At worst case, we perform a finite predefined number of selections  $R_{\text{selec}}$ ; if a consistent set of accepting runs is not found, we proceed with the less efficient centralized procedure in Step 4, which however searches through all sets of all possible accepting runs.
- (4) We create the product  $\tilde{\mathcal{T}}_p = \mathcal{T}_p \otimes \mathcal{A}_p$  where  $\mathcal{A}_p$  is the TBA that accepts all the words that satisfy the formula  $\varphi = \varphi_1 \wedge \dots \wedge \varphi_N$ . An accepting run  $\tilde{r}_p$  of the product is projected into the accepting runs  $\{\tilde{r}_1, \dots, \tilde{r}_N\}$ . If there is no accepting run found in  $\mathcal{T}_p \otimes \mathcal{A}_p$ , then Problem 1 has no solution.
- (5) The abstraction procedure allows to find an explicit feedback law for each transition in  $\mathcal{T}_i$ . Therefore, an accepting run  $\tilde{r}_i^t$  in  $\mathcal{T}_i$  that takes the form of a sequence of transitions is realized in the system in (2) via the corresponding sequence of feedback laws.

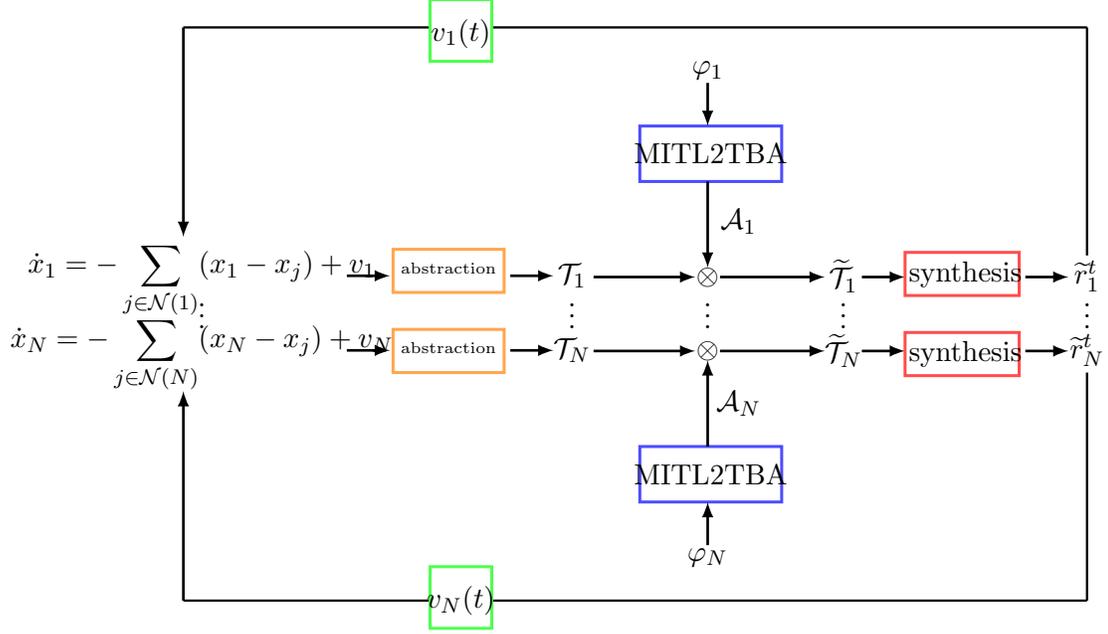


Figure 8: A graphic illustration of the proposed framework.

In order to construct the Buchi WTSs  $\tilde{\mathcal{T}}_p$  and  $\tilde{\mathcal{T}}_i, i \in \mathcal{I}$  that were presented in Steps 2 and 4, consider the following generic definition:

**Definition 11.** Given a WTS  $\mathcal{T}_i = (S_i, S_i^{\text{init}}, Act_i, \longrightarrow_i, d_i, AP_i, L_i)$ , and a TBA  $\mathcal{A}_i = (Q_i, Q_i^{\text{init}}, C_i, Inv_i, E_i, F_i, AP_i, \mathcal{L}_i)$  with  $|C_i|$  clocks and let  $C_i^{\text{max}}$  be the largest constant appearing in  $\mathcal{A}_i$ . Then, their *Büchi WTS*  $\tilde{\mathcal{T}}_i = \mathcal{T}_i \otimes \mathcal{A}_i = (\tilde{S}_i, \tilde{S}_i^{\text{init}}, \tilde{Act}_i, \rightsquigarrow_i, \tilde{d}_i, \tilde{F}_i, AP_i, \tilde{L}_i)$  is defined as follows:

- $\tilde{S}_i \subseteq \{(s_i, q_i) \in S_i \times Q_i : L_i(s_i) = \mathcal{L}_i(q_i)\} \times \mathbb{T}_{\infty}^{|C_i|}$ .
- $\tilde{S}_i^{\text{init}} = S_i^{\text{init}} \times Q_i^{\text{init}} \times \{0\}^{|C_i|}$ .
- $\tilde{Act}_i = Act_i$ .
- $(\tilde{q}, \tilde{act}_i, \tilde{q}') \in \rightsquigarrow_i$  iff
  - $\tilde{q} = (s, q, \nu_1, \dots, \nu_{|C_i|}) \in \tilde{S}_i$ ,
  - $\tilde{q}' = (s', q', \nu'_1, \dots, \nu'_{|C_i|}) \in \tilde{S}_i$ ,
  - $\tilde{act}_i \in Act_i$ ,
  - $(s, \tilde{act}_i, s') \in \longrightarrow_i$ , and
  - there exists  $\gamma, R$ , such that  $(q, \gamma, R, q') \in E_i$ ,  $\nu_1, \dots, \nu_{|C_i|} \models \gamma$ ,  $\nu'_1, \dots, \nu'_{|C_i|} \models Inv_i(q')$ , and for all  $i \in \{1, \dots, |C_i|\}$  it holds that:

$$\nu'_i = \begin{cases} 0, & \text{if } c_i \in R \\ \nu_i + d_i(s, s'), & \text{if } c_i \notin R \text{ and} \\ & \nu_i + d_i(s, s') \leq C_i^{\text{max}} \\ \infty, & \text{otherwise.} \end{cases}$$

Then,  $\tilde{d}_i(\tilde{q}, \tilde{q}') = d_i(s, s')$ .

- $\tilde{F}_i = \{(s_i, q_i, \nu_1, \dots, \nu_{|C_i|}) \in Q_i : q_i \in F_i\}$ .
- $\tilde{L}_i(s_i, q_i, \nu_1, \dots, \nu_{|C_i|}) = L_i(s_i)$ .

The Buchi WTS  $\tilde{\mathcal{T}}_p$  is constructed in a similar way to Def. 11 by using the product of  $\mathcal{T}_p$  and  $\mathcal{A}_p$ . Each Büchi WTS  $\tilde{\mathcal{T}}_i, i \in \mathcal{I}$  is in fact a WTS with a Büchi acceptance condition  $\tilde{F}_i$ . A timed run of  $\tilde{\mathcal{T}}_i$  can be written as  $\tilde{r}_i^t = (q_i(0), \tau_i(0))(q_i(1), \tau_i(1)) \dots$  using the terminology of Def. 5. It is *accepting* if  $q_i(j) \in \tilde{F}_i$  for infinitely many  $i \geq 0$ . An accepting timed run of  $\tilde{\mathcal{T}}_i$  projects onto a timed run of  $\mathcal{T}_i$  that satisfies the local specification formula  $\varphi_i$  by construction. Formally, the following lemma, whose proof follows directly from the construction and the principles of automata-based LTL model checking (see, e.g., [33]), holds:

**Lemma 1.** *Consider an accepting timed run  $\tilde{r}_i^t = (q_i(0), \tau_i(0))(q_i(1), \tau_i(1)) \dots$  of the Büchi WTS  $\tilde{\mathcal{T}}_i$  defined above, where  $q_i(j) = (r_i(j), s_i(j), \nu_{i,1}, \dots, \nu_{i,|C_i|})$  denotes a state of  $\tilde{\mathcal{T}}_i$ , for all  $j \geq 0$ . The timed run  $\tilde{r}_i^t$  projects onto the timed run  $r_i^t = (r_i(0), \tau_i(0))(r_i(1), \tau_i(1)) \dots$ , of the WTS  $\mathcal{T}_i$  that produces the timed word  $w(r_i^t) = (L_i(r_i(0)), \tau_i(0))(L_i(r_i(1)), \tau_i(1)) \dots$  accepted by the TBA  $\mathcal{A}_i$  via its run  $\rho_i = s_i(0)s_i(1) \dots$ . Vice versa, if there exists a timed run  $r_i^t = (r_i(0), \tau_i(0))(r_i(1), \tau_i(1)) \dots$ , of the WTS  $\mathcal{T}_k$  that produces a timed word  $w(r_i^t) = (L_i(r_i(0)), \tau_i(0))(L_i(r_i(1)), \tau_i(1)) \dots$  accepted by the TBA  $\mathcal{A}_i$  via its run  $\rho_i = s_i(0)s_i(1) \dots$  then there exists the accepting timed run  $\tilde{r}_i^t = (q_i(0), \tau_i(0))(q_i(1), \tau_i(1)) \dots$  of  $\tilde{\mathcal{T}}_i$ , where  $q_i(j) = (r_i(j), s_i(j), \nu_{i,1}, \dots, \nu_{i,|C_i|})$ , in  $\tilde{\mathcal{T}}_i$ .*

The proposed framework is depicted in Figure 8. The dynamics (2) of each agent  $i$  is abstracted into a WTS  $\mathcal{T}_i$  (orange rectangles). Then the product between each WTS  $\mathcal{T}_i$  and the TBA  $\mathcal{A}_i$  is computed according to Def. 11. The TBA  $\mathcal{A}_i$  accepts all the words that satisfy the formula  $\varphi_i$  (blue rectangles). For every Büchi WTS  $\tilde{\mathcal{T}}_i$  the controller synthesis procedure that was described in this Section (red rectangles) is performed and a sequence of accepted runs  $\{\tilde{r}_1^t, \dots, \tilde{r}_N^t\}$  is designed. Every accepted run  $\tilde{r}_i^t$  maps into a decentralized controller  $v_i(t)$  which is a solution to Problem 1.

**Proposition 2.** *A solution obtained from Steps 1-5, gives a sequence of controllers  $v_1, \dots, v_N$  that guarantees the satisfaction of the formulas  $\varphi_1, \dots, \varphi_N$  of the agents  $1, \dots, N$  respectively, governed by the dynamics as in (2), thus, they are a solution to Problem 1.*

#### 4.5. Complexity

Denote by  $|\varphi|$  the length of an MITL formula  $\varphi$ . A TBA  $\mathcal{A}_i, i \in \mathcal{I}$  can be constructed in space and time  $2^{\mathcal{O}(|\varphi_i|)}, i \in \mathcal{I}$ . Let  $\varphi_{\max} = \max_{i \in \mathcal{I}} \{|\varphi_i|\}$  be the MITL formula with the longest length. Then, the complexity of Step 1 is  $N2^{\mathcal{O}(|\varphi_{\max}|)}$ . Step 2 costs  $\mathcal{O}(N2^{|\varphi_i|}|\mathcal{S}_i|)$ , where  $|\mathcal{S}_i| = |\hat{\mathbb{I}}|$  is the number of states of the WTS  $\mathcal{T}_i$ . We have the best case complexity as  $\mathcal{O}(NR_{\text{selec}}2^{|\varphi_{\max}|}|\hat{\mathbb{I}}|)$ , since the Step 3 is more efficient than Step 4. The worst case complexity of our proposed framework is when Step 4 is followed, which is  $\mathcal{O}(2^{|\varphi_{\max}|}|\hat{\mathbb{I}}|^N)$ .

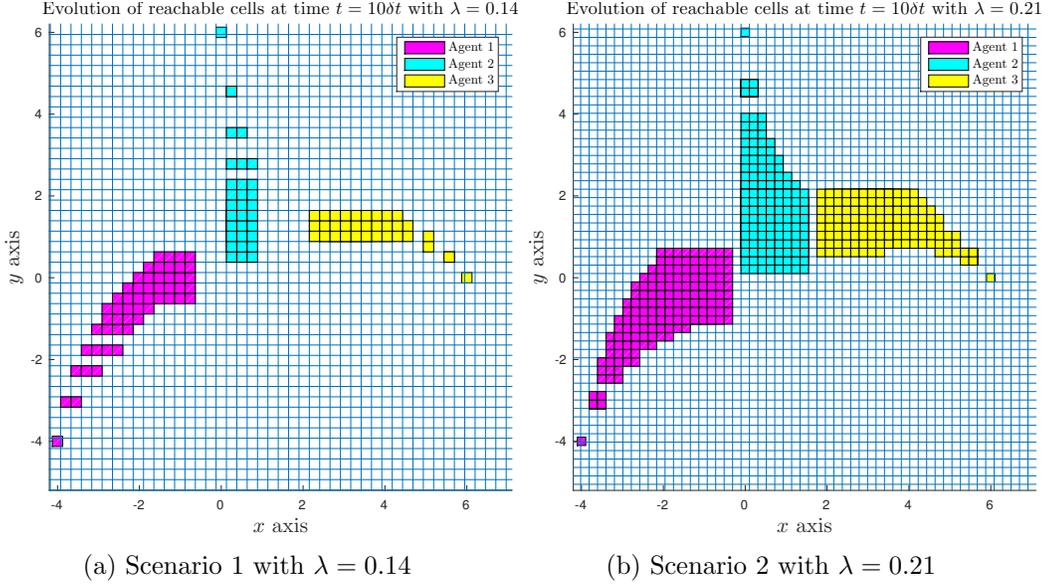


Figure 9.: Two simulation scenarios with  $N = 3$  agents and  $\lambda = 0.14$ ,  $\bar{d}_{\max} = 0.25$  (Figure 9a) and  $\lambda = 0.21$ ,  $\bar{d}_{\max} = 0.20$  (Figure 9b) in a time horizon of  $t = 10\delta t$ .

$N = 3$ agents				
$\lambda = 0.14$			$\lambda = 0.21$	
Step	Reachable States	Time	Reachable States	Time
$\delta t$	2	0.13 sec	24	0.15 sec
$2\delta t$	12	0.05 sec	235	0.11 sec
$3\delta t$	18	0.05 sec	1006	0.88 sec
$4\delta t$	54	1.0 sec	3104	5.17 sec
$5\delta t$	250	0.90 sec	7983	20.80 sec
$6\delta t$	250	1.27 sec	12545	48.77 sec
$7\delta t$	250	1.36 sec	13493	70.21 sec
$8\delta t$	475	1.31 sec	16078	74.56 sec
$9\delta t$	875	2.21 sec	23690	107.79 sec
$10\delta t$	1100	3.09 sec	33171	185.91 sec
		Total Time: 11.41 sec	Total Time: 514.37 sec	

Table 1.: This table shows the simulation statistics of an example with  $N = 3$  agents whose reachable cells are depicted in Figure 9. The first column is the number of the time step (recall that the duration of each transition in the WTS is exactly  $\delta t$ ). The other columns show for  $\lambda = 0.14$  and  $\lambda = 0.21$ , the number of the reachable states generated by the product WTS  $\mathcal{T}_p$ , as given in Definition 8 of the revised version, as well as the required computational time. The total time stands for the computation of the reachable states.

## 5. Simulation Results

In order to show how the proposed framework scales with respect to the number of agents and the time solution horizon, we consider two simulation examples with two simulation scenarios each.

**Simulation Example 1 :** Consider a system of three agents with  $x_i \in \mathbb{R}^2$ ,  $i \in \mathcal{I} = \{1, 2, 3\}$ ,  $\mathcal{N}(1) = \{2\} = \mathcal{N}(3)$ ,  $\mathcal{N}(2) = \{1, 3\}$  is considered. According to (2), the dynamics are given as:  $\dot{x}_1 = -(x_1 - x_2) + v_1$ ,  $\dot{x}_2 = -(x_2 - x_1) - (x_2 - x_3) + v_2$  and  $\dot{x}_3 = -(x_3 - x_2) + v_3$ . The simulation parameters are set to  $L_1 = \sqrt{2}$ ,  $L_2 = 2$  and  $\delta t = 0.1$ . The initial agents' positions are set to  $(-4, 4)$ ,  $(0, 6)$  and  $(7, 0)$  respectively. We consider Scenario 1 and Scenario 2 with  $\lambda = 0.14$ ,  $\bar{d}_{\max} = 0.25$  and  $\lambda = 0.21$ ,  $\bar{d}_{\max} = 0.20$ , as is depicted in Figure 9a and Figure 9b, respectively. The cell decomposition presented in this paper and the reachable cells of each agent are depicted in Figure 9a-9b. The reachable cells of each agent are depicted with purple, cyan and yellow respectively. In Figure 9a-9b we can observe the evolution of the reachable sets of each agent at time  $t = 10\delta t$ . It can be observed that the agents are not necessarily moving between neighboring cells and not all the individual runs satisfy the desired specification. The simulation statistics are depicted in Table 1. The simulations were carried out in MATLAB Environment on a desktop with 8 cores, 3.60GHz CPU and 16GB of RAM.

**Simulation Example 2 :** Consider a multi-agent system with  $x_i \in \mathbb{R}^2$ ,  $i \in \mathcal{I} = \{1, 2, 3, 4\}$ ,  $\mathcal{N}(1) = \{2\}$ ,  $\mathcal{N}(2) = \{1, 3\}$ ,  $\mathcal{N}(3) = \{2, 4\}$ ,  $\mathcal{N}(4) = \{3\}$ . According to (2), the dynamics are given as:  $\dot{x}_1 = -(x_1 - x_2) + v_1$ ,  $\dot{x}_2 = -(x_2 - x_1) - (x_2 - x_3) + v_2$ ,  $\dot{x}_3 = -(x_3 - x_2) - (x_2 - x_4) + v_3$  and  $\dot{x}_4 = -(x_4 - x_3) + v_4$ . The simulation parameters are set to  $L_1 = \sqrt{2}$ ,  $L_2 = 2$ ,  $\delta t = 0.1$ . The workspace is decomposed into square cells, which are depicted with blue color in Figure 10. The initial agents' positions are set to  $(-4, 4)$ ,  $(0, 6)$ ,  $(7, 0)$  and  $(4, -5)$ , respectively. We consider Scenario 1 and Scenario 2 with  $\lambda = 0.14$ ,  $\bar{d}_{\max} = 0.25$  and  $\lambda = 0.21$ ,  $\bar{d}_{\max} = 0.20$ , as is depicted in Figure 9a and Figure 9b, respectively. The specification formulas for the Scenario 1 are set to  $\varphi_1 = \diamond_{[0.2, 1.0]} \{\text{green}\}$ ,  $\varphi_2 = \diamond_{[0.1, 1.0]} \{\text{orange}\}$ ,  $\varphi_3 = \diamond_{[0.5, 1.5]} \{\text{black}\}$  and  $\varphi_4 = \diamond_{[0.3, 2.3]} \{\text{gray}\}$ , respectively. The cell decomposition presented in this paper, the reachable cells of each agent up to time  $t = 10\delta t$  and the goal regions are depicted in Figure 10. The reachable cells of each agent are depicted with purple, cyan and yellow respectively. Note that the agents' transitions are not necessarily performed between neighboring cells. The individual consistent runs  $\tilde{r}_1^t$ ,  $\tilde{r}_2^t$ ,  $\tilde{r}_3^t$  and  $\tilde{r}_4^t$  of agents 1, 2, 3 and 4 that satisfy the formulas  $\varphi_1$ ,  $\varphi_2$ ,  $\varphi_3$  and  $\varphi_4$ , respectively are depicted in Figure 10 with black arrows. Each arrow represents a transition from a state to another according to Def. 8. The product WTS  $\mathcal{T}_p$  has 52877 reachable states in case of  $\lambda = 0.14$  and 1255547 when  $\lambda = 0.21$  since, according to Section 4.2.2, larger values of  $\lambda$  lead to finer discretization. Agent 1 satisfies  $\varphi_1$  in  $5\delta t$ , agent 2 satisfies  $\varphi_2$  in  $2\delta t$ , agent 3 satisfies  $\varphi_3$  in  $8\delta t$  and agent 4 satisfies  $\varphi_4$  in  $10\delta t$ . The simulation is performed in a horizon of 10 steps and it takes 668.80 sec (609.5 sec for the abstraction and 59.30 sec for the graph search) for Scenario 1.

## 6. Conclusions and Future Work

A systematic method for controller synthesis of dynamically coupled multi-agent path-planning has been proposed, in which timed constraints of fulfilling a high-level specification are imposed to the system. The solution involves a boundedness analysis, the abstraction of each agent's motion into WTSs, TBAs as well as Büchi WTSs construction. The simulation example demonstrates our solution approach. Future work includes further computational improvement of the abstraction method and more complicated high-level tasks being imposed to the agents in order to exploit the expressiveness of MITL formulas.

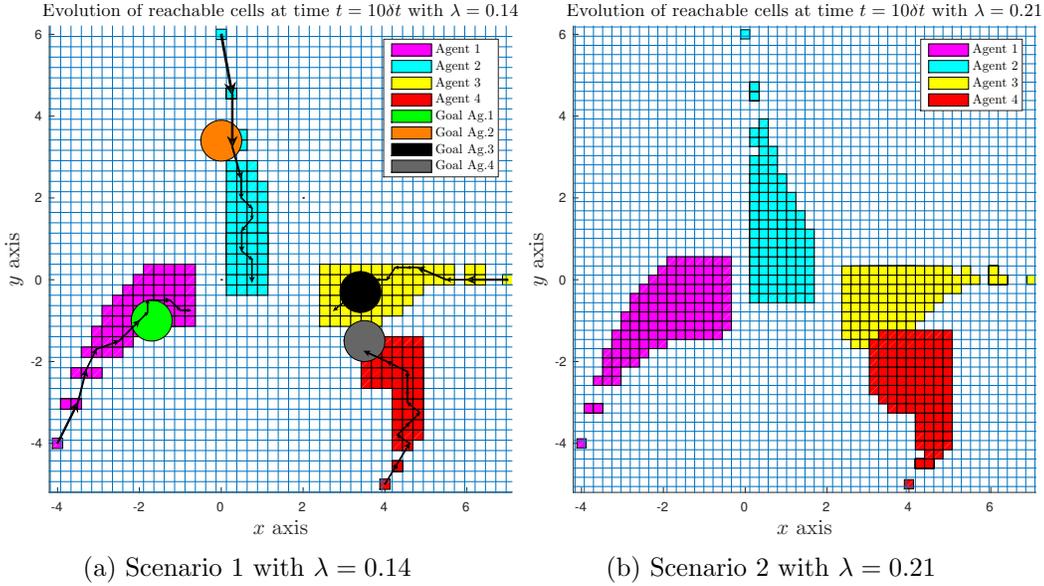


Figure 10.: Two simulation scenarios with  $N = 3$  agents and  $\lambda = 0.14$ ,  $\bar{d}_{\max} = 0.25$  (Figure 10a) and  $\lambda = 0.21$ ,  $\bar{d}_{\max} = 0.20$  (Figure 10b) in a time horizon of  $t = 10\delta t$ . The figure depicts the evolution of the agents' reachable cells up to time  $t = 10\delta t$ . The scenario is the same with the simulation of Section 6 of the revised paper, but now we have removed agent 4. The rest of agents start from the same initial conditions as in the previous Scenarios.

$N = 4$ agents				
$\lambda = 0.14$			$\lambda = 0.21$	
Step	Reachable States	Time	Reachable States	Time
$\delta t$	8	0.22 sec	24	0.32 sec
$2\delta t$	39	0.11 sec	208	0.24 sec
$3\delta t$	216	0.16 sec	6702	5.92 sec
$4\delta t$	1610	17.97 sec	26843	74.95 sec
$5\delta t$	3168	16.73 sec	89817	91.68 sec
$6\delta t$	5346	21.87 sec	133904	252.12 sec
$7\delta t$	5808	31.55 sec	222037	451.61 sec
$8\delta t$	10168	37.39 sec	358941	1427.65 sec
$9\delta t$	23004	93.98 sec	644489	4803.12 sec
$10\delta t$	52877	389.50 sec	1255547	19321.07 sec
Total Time: 609.50 sec			Total Time: 26429 sec	

Table 2.: This table shows the simulation statistics of an example with  $N = 4$  agents whose reachable cells are depicted in Figure 10. The number of reachable states of each time step is depicted. The total simulation time stands for the abstraction procedure.

## References

- [1] T. Wongpiromsarn, U. Topcu, and R. Murray. Receding Horizon Control for Temporal Logic Specifications. *13th ACM International Conference on Hybrid Systems: Computation and Control (HSCC)*, pages 101–110, 2010.

- [2] M. Guo and D. Dimarogonas. Multi-Agent Plan Reconfiguration Under Local LTL Specifications. *The International Journal of Robotics Research*, 34(2):218–235, 2015.
- [3] Y. Kantaros and M. Zavlanos. A Distributed LTL-Based Approach for Intermittent Communication in Mobile Robot Networks. *American Control Conference (ACC)*, pages 5557–5562, 2016.
- [4] I. Saha, R. Ramaithitima, V. Kumar, G. Pappas, and S. Seshia. Implan: Scalable Incremental Motion Planning for Multi-Robot Systems. *2016 ACM/IEEE 7th International Conference on Cyber-Physical Systems (ICCP)*, 2016.
- [5] J. Liu and P. Prabhakar. Switching Control of Dynamical Systems from Metric Temporal Logic Specifications. *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5333–5338, 2014.
- [6] V. Raman, A. Donzé, D. Sadigh, R. Murray, and S. Seshia. Reactive Synthesis from Signal Temporal Logic Specifications. *18th International Conference on Hybrid Systems: Computation and Control (HSCC)*, pages 239–248, 2015.
- [7] J. Fu and U. Topcu. Computational Methods for Stochastic Control with Metric Interval Temporal Logic Specifications. *54th IEEE Conference on Decision and Control (CDC)*, pages 7440–7447, 2015.
- [8] Y. Zhou, D. Maity, and John S. Baras. Timed Automata Approach for Motion Planning Using Metric Interval Temporal Logic. *European Control Conference (ECC)*, 2016.
- [9] S. Karaman and E. Frazzoli. Vehicle Routing Problem with Metric Temporal Logic Specifications. *47th IEEE Conference on Decision and Control (CDC 2008)*, pages 3953–3958, 2008.
- [10] A. Nikou, J. Tumova, and D. Dimarogonas. Cooperative Task Planning of Multi-Agent Systems Under Timed Temporal Specifications. *American Control Conference (ACC)*, pages 13–19, 2016.
- [11] R. Alur, T. Henzinger, G. Lafferriere, and G. Pappas. Discrete Abstractions of Hybrid Systems. *Proceedings of the IEEE*, 88(7):971–984, 2000.
- [12] M. Zamani, G. Pola, M. Mazo, and P. Tabuada. Symbolic Models for Nonlinear Control Systems without Stability Assumptions. *IEEE Transactions on Automatic Control (TAC)*, 57(7), 2012.
- [13] M. Zamani, M. Mazo, and A. Abate. Finite Abstractions of Networked Control Systems. *53rd IEEE Conference on Decision and Control (CDC)*, pages 95–100, 2014.
- [14] P. Meyer, A. Girard, and E. Witrant. Compositional Abstraction and Safety Synthesis Using Overlapping Symbolic Models. *IEEE Transaction on Automatic Control (TAC)*, 2017.
- [15] O. Hussein, A. Ames, and P. Tabuada. Abstracting Partially Feedback Linearizable Systems Compositionally. *IEEE Control Systems Letters*, vol. 1, no. 2, pp. 227–232, Oct. 2017.
- [16] D. Boskos and D. Dimarogonas. Decentralized Abstractions For Multi-Agent Systems Under Coupled Constraints. *54th IEEE Conference on Decision and Control (CDC)*, pages 282–287, 2015.
- [17] R. Horn and C. Johnson. *Topics in Matrix Analysis*. Cambridge University Press, 1994.
- [18] M. Mesbahi and M. Egerstedt. *Graph Theoretic Methods in Multiagent Networks*. 2010.
- [19] R. Alur and D.L. Dill. A Theory of Timed Automata. *Theoretical Computer Science*, 126(2):183–235, 1994.

- [20] D. D. Souza and P. Prabhakar. On the Expressiveness of MTL in the Pointwise and Continuous Semantics. *International Journal on Software Tools for Technology Transfer*, 9(1):1–4, 2007.
- [21] J. Ouaknine and J. Worrell. On the Decidability of Metric Temporal Logic. *20th Annual IEEE Symposium on Logic in Computer Science (LICS)*, pages 188–197, 2005.
- [22] R. Alur, T. Feder, and T. A. Henzinger. The Benefits of Relaxing Punctuality. *Journal of the ACM (JACM)*, 43(1):116–146, 1996.
- [23] M. Reynolds. Metric Temporal Logics and Deterministic Timed Automata. 2010.
- [24] P. Bouyer. From Qualitative to Quantitative Analysis of Timed Systems. *Mémoire dhabilitation, Université Paris*, 7:135–175, 2009.
- [25] S. Tripakis. Checking Timed Buchi Automata Emptiness on Simulation Graphs. *ACM Transactions on Computational Logic (TOCL)*, 10(3), 2009.
- [26] O. Maler, D. Nickovic, and A. Pnueli. From MITL to Timed Automata. *International Conference on Formal Modeling and Analysis of Timed Systems*, pages 274–289, 2006.
- [27] D. Ničković and N. Piterman. From MTL to Deterministic Timed Automata. *Formal Modeling and Analysis of Timed Systems*, 2010.
- [28] T. Brihaye, G. Geeraerts, H. Ho, and B. Monmege. MightyL: A Compositional Translation from MITL to Timed Automata. *29th International Conference on Computer Aided Verification (CAV)*, 2017.
- [29] S. Karaman and E. Frazzoli. Linear Temporal Logic Vehicle Routing with Applications to Multi-UAV Mission Planning. *International Journal of Robust and Nonlinear Control*, 21(12):1372–1395, 2011.
- [30] M. Kloetzer and C. Belta. Automatic Deployment of Distributed Teams of Robots From Temporal Motion Specifications. *IEEE Transactions on Robotics*, 26(1):48–61, 2010.
- [31] M. Kloetzer, X. C. Ding, and C. Belta. Multi-Robot Deployment from LTL Specifications with Reduced Communication. *50th IEEE Conference on Decision and Control (CDC)*, pages 4867–4872, 2011.
- [32] P. Tabuada. *Verification and Control of Hybrid Systems: a Symbolic Approach*. Springer Science and Business Media, 2009.
- [33] C. Baier, J.P. Katoen, and K. G. Larsen. *Principles of Model Checking*. MIT Press, 2008.
- [34] D. Liberzon. *Switching in Systems and Sontrol*. Springer Science and Business Media, 2012.
- [35] D. Boskos and D. Dimarogonas. Robust Connectivity Analysis for Multi-Agent Systems. *54th IEEE Conference on Decision and Control (CDC)*, pages 6767–6772, 2015.
- [36] Miroslav M. Fiedler. Algebraic Connectivity of Graphs. *Czechoslovak Mathematical Journal*, 23(2):298–305, 1973.

## Appendix A. Proof of Theorem 1

Consider the following candidate Lyapunov function  $V : \mathbb{R}^{Nn} \rightarrow \mathbb{R}_{\geq 0}$

$$V(x) = \frac{1}{2} \sum_{i=1}^N \sum_{j \in \mathcal{N}(i)} \|x_i - x_j\|^2 = \|\tilde{x}\|^2 > 0.$$

The time derivative of  $V$  along the trajectories of (2), can be computed as

$$\begin{aligned}\dot{V} &= [\nabla V(x)]^\top \dot{x} \\ &= \sum_{k=1}^n \left\{ c \left( \frac{\partial V}{\partial x}, k \right)^\top [-L(\mathcal{G}) c(x, k) + c(v, k)] \right\},\end{aligned}\quad (\text{A1})$$

where  $c \left( \frac{\partial V}{\partial x}, k \right) = \left[ \frac{\partial V}{\partial x_1^k} \quad \dots \quad \frac{\partial V}{\partial x_N^k} \right]^\top$ . By computing the partial derivative of the Lyapunov function with respect to vector  $x_i$ ,  $i \in \mathcal{I}$  we get  $\frac{\partial V}{\partial x_i} = \sum_{j \in \mathcal{N}(i)} (x_i - x_j)$ ,  $i \in \mathcal{I}$  from which we have that  $c \left( \frac{\partial V}{\partial x}, k \right)^\top = c(x, k)^\top L(\mathcal{G})$ ,  $k = 1, \dots, n$ . Thus, by substituting the last in (A1) we get

$$\begin{aligned}\dot{V} &= - \sum_{k=1}^n \left\{ c(x, k)^\top [L(\mathcal{G})]^2 c(x, k) \right\} + \sum_{k=1}^n \left\{ c(x, k)^\top [L(\mathcal{G})]^2 c(v, k) \right\} \\ &\leq - \sum_{k=1}^n \left\{ c(x, k)^\top [L(\mathcal{G})]^2 c(x, k) \right\} + \left\| \sum_{k=1}^n \left\{ c(x, k)^\top L(\mathcal{G}) c(v, k) \right\} \right\|.\end{aligned}\quad (\text{A2})$$

For the first term of (A2) we have that

$$\sum_{k=1}^n \left\{ c(x, k)^\top L(\mathcal{G})^2 c(x, k) \right\} = \sum_{k=1}^n \|L(\mathcal{G}) c(x, k)\|^2.$$

For the second term of (A2) we get

$$\begin{aligned}\left\| \sum_{k=1}^n \left\{ c(x, k)^\top L(\mathcal{G}) c(v, k) \right\} \right\| &= \left\| \sum_{k=1}^n \left\{ c(x, k)^\top D(\mathcal{G}) D(\mathcal{G})^\top c(v, k) \right\} \right\| \\ &\leq \sum_{k=1}^n \left\{ \left\| D(\mathcal{G})^\top c(x, k) \right\| \left\| D(\mathcal{G})^\top \right\| \|c(v, k)\| \right\} \\ &= \left\| D(\mathcal{G})^\top \right\| \sum_{k=1}^n \{ \|c(\tilde{x}, k)\| \|c(v, k)\| \}.\end{aligned}\quad (\text{A3})$$

By using the Cauchy-Schwarz inequality in (A3) we get

$$\begin{aligned}\left\| \sum_{k=1}^n \left\{ c(x, k)^\top L(\mathcal{G}) c(v, k) \right\} \right\| &\leq \left\| D(\mathcal{G})^\top \right\| \left( \sum_{k=1}^n \|c(\tilde{x}, k)\|^2 \right)^{\frac{1}{2}} \left( \sum_{k=1}^n \|c(v, k)\|^2 \right)^{\frac{1}{2}} \\ &= \left\| D(\mathcal{G})^\top \right\| \|\tilde{x}\| \|v\| \leq \left\| D(\mathcal{G})^\top \right\| \|\tilde{x}\| \sqrt{N} \|v\|_\infty,\end{aligned}$$

where  $\|v\|_\infty = \max \{ \|v_i\| : i \in \mathcal{I} \} \leq v_{\max}$ . Thus, by combining the previous inequali-

ties, (A2) is written

$$\dot{V} \leq - \sum_{k=1}^n \left\{ \|L(\mathcal{G})c(x, k)\|^2 \right\} + \sqrt{N} \left\| D(\mathcal{G})^\top \right\| \|\tilde{x}\| v_{\max}. \quad (\text{A4})$$

By exploiting Lemma 2, (A4) is written as:

$$\begin{aligned} \dot{V} &\leq -\lambda_2^2(\mathcal{G}) \sum_{k=1}^n \left\{ \|c(x^\perp, k)\|^2 \right\} + \sqrt{N} \left\| D(\mathcal{G})^\top \right\| \|\tilde{x}\| v_{\max} \\ &= -\lambda_2^2(\mathcal{G}) \|x^\perp\|^2 + \sqrt{N} \left\| D(\mathcal{G})^\top \right\| \|\tilde{x}\| v_{\max} \\ &\leq -\frac{\lambda_2^2(\mathcal{G})}{2(N-1)} \|\tilde{x}\|^2 + \sqrt{N} \left\| D(\mathcal{G})^\top \right\| \|\tilde{x}\| v_{\max} \\ &\leq -K_1 \|\tilde{x}\| (\|\tilde{x}\| - K_2 v_{\max}). \end{aligned} \quad (\text{A5})$$

where  $K_1 = \frac{\lambda_2^2(\mathcal{G})}{2(N-1)} > 0$ . By using the following implication  $\tilde{x} = D^\top(\mathcal{G})x \Rightarrow \|\tilde{x}\| = \|D(\mathcal{G})^\top x\| \leq \|D(\mathcal{G})^\top\| \|x\|$ , apparently, we have that  $0 < V(x) = \|\tilde{x}\|^2 \leq \|D(\mathcal{G})^\top\|^2 \|x\|^2$  and  $\dot{V}(x) < 0$  when  $\|\tilde{x}\| \geq \bar{R} > K_2 v_{\max}$ . Thus, there exists a finite time  $T > 0$  such that the trajectory will enter the compact set  $\mathcal{X} = \{x \in \mathbb{R}^{Nn} : \|\tilde{x}\| \leq \bar{R}\}$  and remain there for all  $t \geq T$  with  $\bar{R} > K_2 v_{\max}$ . This can be extracted from the following. Let us define the compact set

$$\Omega = \{x \in \mathbb{R}^{Nn} : K_2 v_{\max} < \bar{R} \leq \|\tilde{x}\| \leq \bar{M}\},$$

where  $\bar{M} = V(x(0)) = \|\tilde{x}(0)\|^2$ . Without loss of generality it is assumed that it holds  $\bar{M} > \bar{R}$ . Let us define the compact sets  $\mathcal{S}_1 = \{x \in \mathbb{R}^{Nn} : \|\tilde{x}\| \leq \bar{M}\}$ ,  $\mathcal{S}_2 = \{x \in \mathbb{R}^{Nn} : \|\tilde{x}\| \leq K_2 v_{\max}\}$ . From the equivalences  $\forall x \in \mathcal{S}_1 \Leftrightarrow V(x) = \|\tilde{x}\|^2 \leq \bar{M}^2, \forall x \in \mathcal{S}_2 \Leftrightarrow V(x) = \|x\|^2 \leq K_2^2 v_{\max}^2$ , we have that the boundaries  $\partial\mathcal{S}_1, \partial\mathcal{S}_2$  of sets  $\mathcal{S}_1, \mathcal{S}_2$  respectively, are two level sets of the Lyapunov function  $V$ . By taking the above into consideration we have that  $\partial\mathcal{S}_2 \subsetneq \partial\mathcal{S}_1$ . Hence, we get from (A5) that:

$$\dot{V}(x) < 0, \forall x \in \Omega = \mathcal{S}_1 \setminus \mathcal{S}_2, \quad (\text{A6})$$

In view of (A6) and the fact that the sets  $\mathcal{S}_1, \mathcal{S}_2$  are defined in terms of level sets of  $V$ , we conclude that both  $\mathcal{S}_1$  and  $\mathcal{S}_2$  are invariant with respect to the system (2). Consequently, according to [34, Lemma 5.1] the trajectory that starts inside the set  $\mathcal{S}_1$  has to enter the interior of the set of  $\mathcal{S}_2$  in finite time  $T > 0$  and remain there for all time  $t \geq T$ .

## Appendix B. Proof of Lemma 2

**Lemma 2.** *Let  $x^\perp$  be the projection of the vector  $x \in \mathbb{R}^{Nn}$  to the orthogonal complement of the subspace  $H = \{x \in \mathbb{R}^{Nn} : x_1 = \dots = x_N\}$ . Then, the following properties*

hold:

$$\begin{aligned} \|L(\mathcal{G}) c(x, k)\| &\geq \lambda_2(\mathcal{G}) \|c(x^\perp, k)\|, \quad \forall k \in \mathcal{I}, \\ \|x^\perp\| &\geq \frac{1}{\sqrt{2(N-1)}} \|\tilde{x}\|. \end{aligned}$$

**Proof.** The proof can be found in [35, Appendix A].  $\square$

### Appendix C. Sufficient Conditions for Well-Possessedness of the Abstraction

We investigate here if the system (2) satisfies the sufficient conditions **(C1)**-**(C3)** for well-posed abstractions.

**(C1)** For every  $i \in \mathcal{I}, \forall x \in \mathbb{R}^{Nn} : \tilde{x} \in \mathcal{X}$  and  $\text{pr}_i(x) = (x_i, \mathbf{x}_j)$  it holds that:

$$\begin{aligned} \|f_i(x_i, \mathbf{x}_j)\| &= \left\| - \sum_{j \in \mathcal{N}(i)} (x_i - x_j) \right\| \leq \sum_{j \in \mathcal{N}(i)} \|x_i - x_j\| \\ &\leq \sum_{(i,j) \in \mathcal{E}} \|x_i - x_j\| = \Delta x \leq \bar{R}. \end{aligned}$$

Thus,  $M = \bar{R}$ . We have also that  $\|D(\mathcal{G})^\top\| = \sqrt{\lambda_{\max}(D(\mathcal{G})D(\mathcal{G})^\top)} = \sqrt{\lambda_{\max}(\mathcal{G})}$  and  $\lambda_2(\mathcal{G}) \leq \frac{N}{N-1} \min\{N_i : i \in \mathcal{I}\}$  from [36]. For  $N > 2$  it holds that  $\lambda_2(\mathcal{G}) < N$ . From Theorem (1) we have that  $\bar{R} > K_2 v_{\max} \Leftrightarrow M > K_2 v_{\max}$ . It holds that  $M > v_{\max}$  since

$$\begin{aligned} K_2 &= \frac{2\sqrt{N}(N-1) \|D(\mathcal{G})^\top\|}{\lambda_2^2(\mathcal{G})} = \frac{2\sqrt{N}(N-1) \sqrt{\lambda_{\max}(\mathcal{G})}}{\sqrt{\lambda_2^3(\mathcal{G})} \sqrt{\lambda_2(\mathcal{G})}} \\ &\geq \frac{2\sqrt{N}(N-1)}{\sqrt{N^3}} \sqrt{\frac{\lambda_{\max}(\mathcal{G})}{\lambda_2(\mathcal{G})}} \geq \frac{2\sqrt{N}(N-1)}{\sqrt{N^3}} > 1. \end{aligned}$$

**(C2)** Starting from the left hand side of (7) we get:

$$\begin{aligned} \|f_i(x_i, \mathbf{x}_j) - f_i(x_i, \mathbf{y}_j)\| &= \left\| - \sum_{j \in \mathcal{N}(i)} (x_i - x_j) + \sum_{j \in \mathcal{N}(i)} (x_i - y_j) \right\| \\ &\leq \max\{\sqrt{N_i} : i \in \mathcal{I}\} \|(x_i, \mathbf{x}_j) - (x_i, \mathbf{y}_j)\|. \end{aligned}$$

Thus, the condition **(C2)** holds and the Lipschitz constant is  $L_1 = \max\{\sqrt{N_i} : i \in \mathcal{I}\} > 0$ , where the inequality  $\left(\sum_{i=1}^{\rho} \alpha_i\right)^2 \leq \rho \left(\sum_{i=1}^{\rho} \alpha_i^2\right)$  is used.

**(C3)** By using the same methodology as in **(C2)**, we conclude that  $L_2 = \max\{N_i : i \in \mathcal{I}\} > 0$ .